UNIVERSIDAD CARLOS III DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

INGENIERÍA TÉCNICA EN INFORMÁTICA DE GESTIÓN

PROYECTO FIN DE CARRERA

– IMPLEMENTACIÓN DE UN SERVIDOR DE COMUNICACIONES ENTRE LOS DISTINTOS MÓDULOS DE UN SISTEMA DE GESTIÓN TURÍSTICA –

Autor: D. Javier Enrique Paniagua Sánchez.

Tutor: D. Ángel García Olaya.

AGRADECIMIENTOS

Doy las gracias a todas las personas que me han apoyado en la realización de este proyecto y a lo largo de mi vida estudiantil. En especial agradecer a mis padres su confianza y apoyo, tanto en los buenos momentos como en las etapas más bajas. Siempre han intentado inculcarme fidelidad con mis gustos e ideas, a la hora de elegir mi carrera y un sentido de responsabilidad y constancia para llevarla a buen fin. Cuando los he necesitado han estado ahí para aconsejarme, pero intentando ser imparciales y no influir, a fin de que la decisión siempre fuera mía. Gracias por vuestro apoyo y cariño.

También quisiera agradecer a mis hermanos el apoyo y paciencia que han tenido; sé, que para ellos (ya que son adolescentes) a veces era duro renunciar a jugar o chatear con sus amigos, porque yo estaba utilizando el ordenador. Después, de todo se que están felices de que haya llegado hasta aquí en mis estudios.

Doy un agradecimiento especial a mis abuelos por su apoyo y comprensión al no haber podido visitarlos durante más tiempo este verano, ellos saben bien que a veces hay que sacrificarse para conseguir lo que deseamos.

Quisiera también mencionar a mis amigos y compañeros de estudios. Todos hemos colaborado en tareas, practicas… aportando nuestras ideas y conocimientos para que su realización fuera exitosa, y no nos rindiéramos nunca.

Por último agradecer al tutor de este proyecto, Ángel García Olaya, su orientación, guía y colaboración en la búsqueda de información y resolución de los problemas surgidos durante el mismo.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1. INTRODUCCIÓN

1.1. Marco del proyecto de fin de carrera

El proyecto de fin de carrera (PFC), "Implementación de un servidor de comunicaciones entre los distintos módulos de un sistema de gestión turística", que se va a desarrollar está promovido por el gran auge que se ha producido en los últimos años en el terreno de la tecnología. Esto ha provocado también un rápido avance en el mundo de la informática.

Estos avances tanto en la tecnología como en la informática, han propiciado que el desarrollo de software no sea exclusivo para ordenadores (de sobremesa o portátiles) sino también para dispositivos móviles (teléfonos móviles, PDA, Smartphone, TabletPc,…) y otros dispositivos (videoconsolas,…).

Aunque los dispositivos móviles, debido a sus limitaciones en el hardware, no pueden gestionar aplicaciones que requieran un procesamiento muy elevado, sí puede desarrollarse software con ciertas limitaciones.

El PFC que se va a llevar a cabo, está relacionado con el desarrollo de aplicaciones, tanto en un ordenador de sobremesa, como en dispositivos móviles.

Todos estos avances en el ámbito de la tecnología han provocado la aparición de nuevos protocolos de comunicación, para comunicar los dispositivos móviles entre sí o con otros dispositivos diferentes (por ejemplo, ordenadores de sobremesa), como son: WIFI (Wireless Fidelity), GPRS (General Packet Radio Service), BLUETOOTH, IrDA (Infrared Data Association)…, algunos de éstos serán los medios de comunicación que se utilizarán en la implementación el proyecto.

Los protocolos de comunicación que hemos mencionado anteriormente son protocolos de comunicación inalámbricos, es decir, no necesitan de forma directa el empleo de cables. Esta propiedad de los protocolos de comunicación inalámbricos nos resultará muy útil a la hora de realizar comunicaciones entre los dispositivos móviles y el resto de dispositivos.

Pero las comunicaciones inalámbricas no son el único mecanismo de comunicación entre dichos dispositivos, también es posible realizar la comunicación mediante el cable USB (Universal Serial Bus).

Todo lo expuesto nos va ser de gran utilidad a la hora de realizar el proyecto, ya que éste consiste en el desarrollo de un módulo de comunicaciones, implementado tanto en un

dispositivo móvil (en este caso será en una PDA, Personal Digital Assistant), como en un ordenador de sobremesa.

Para poder presentar los objetivos generales del proyecto, en primer lugar daremos una visión general del sistema en el que se va desplegar el módulo de comunicación, aunque se comentará con más detalle en el apartado 2.1.5.

El sistema en el que se va a desplegar este módulo de comunicaciones es un sistema encargado de gestionar información de interés turístico, es decir, es capaz de ofrecer a los clientes mapas de una zona turística, un lista de monumentos o lugares de interés de dicha zona o incluso realizar planificaciones de rutas.

Debido a la alta complejidad del sistema, se encuentra dividido en varios módulos o proyectos, los cuales son:

- Aplicación de usuario en el dispositivo móvil (PDA).
- Módulo de comunicación, estará desplegado en el dispositivo móvil y en el servidor del sistema.
- Módulo de gestión de mapas.
- Módulo de gestión de monumentos.
- Planificador de visitas.
- Módulo de gestión de transportes.

En el siguiente apartado se expondrá una breve descripción de los alcances y objetivos generales que tiene el proyecto fin de carrera.

1.2. Alcance y objetivos generales

Después de realizar una breve introducción al sistema en el que se va a desplegar el módulo de comunicación que se va a desarrollar en el proyecto, procedemos a hacer una pequeña descripción del módulo de comunicación y a indicar cuáles son los objetivos generales de este proyecto fin de carrera.

El módulo de comunicación es una parte fundamental del sistema, ya que permite la comunicación entre los módulos restantes. Este módulo está formado a su vez por dos sub-módulos, uno desplegado en la aplicación del dispositivo móvil y el otro en el servidor del sistema.

Podríamos decir que el objetivo principal del proyecto es ser capaces de permitir una comunicación entre todos los módulos que componen el sistema, pero principalmente entre la

aplicación de usuario, que reside en el dispositivo móvil, y los módulos que reciben las peticiones del cliente.

Otros requisitos, no menos importantes son:

- La comunicación entre la aplicación de usuario y los módulos que reciben las peticiones debe ser fluida.
- La comunicación entre la aplicación de usuario y el módulo de comunicación de la parte del servidor del sistema debe de realizarse mediante una conexión segura, debido a que se intercambia información confidencial del cliente.

Más adelante se dará una explicación más detalla y gráfica tanto del sistema al que pertenece el módulo de comunicación como del propio módulo.

1.3. Estructura del documento

En este apartado se va a exponer el contenido de los distintos capítulos que forman parte de este documento.

Capítulo 2, Estado de la cuestión. En esta sección del documento hablaremos sobre los estudios relacionados con nuestro proyecto (como la estructura cliente-servidor, los protocolos de comunicación,…) y las tecnologías y herramientas utilizadas para el desarrollo del proyecto.

También haremos una breve revisión del estado en el que se encuentran las tecnologías empleadas (como el desarrollo de aplicaciones para dispositivos, la existencia de librerías en el empleo de conexiones inalámbricas, el estado de desarrollo de la estructura cliente-servidor,…).

Capítulo 3, Objetivos del proyecto de fin de carrera. En este apartado nos centraremos en explicar los objetivos que tendrá que cumplir el proyecto.

Capítulo 4, Memoria del trabajo realizado. Este capítulo es el más amplio del documento, ya que explica todo aquello que se ha llevado a cabo para la realización del proyecto, como pueden ser las descripciones de las funcionalidades más importantes, el porqué de la solución implementada,…, y además un conjunto de diagramas que nos ayudarán a entender mejor la implementación del módulo de comunicación.

Capítulo 5, Resultados. Esta sección recogerá toda aquella información necesaria para demostrar el correcto funcionamiento del módulo de comunicación desarrollado en el proyecto.

Capítulo 6, Conclusiones. En este capítulo se encuentran detalladas las principales conclusiones que se han obtenido de la realización del proyecto.

Capítulo 7, Futuras líneas de trabajo. Este apartado se centra en describir cuáles podrían ser las futuras actualizaciones que podría sufrir el módulo desarrollado, tanto en las funcionalidades implementadas, como en la optimización del mismo.

Capítulo 8, Bibliografía. En este último capítulo podemos ver qué recursos hemos utilizado para la realización del proyecto de fin de carrera.

Anexo A, Instalación y configuración de las herramientas de desarrollo. En este anexo se mostrarán los procedimientos de instalación y configuración de las herramientas que utilizaremos en el desarrollo del proyecto.

Anexo B, Manual de usuario. En este anexo se realizará una descripción detalla de cómo utilizar la aplicación.

CAPÍTULO 2.
ESTADO DE LA CUESTIÓN

2.1. Estado actual

Dentro de este apartado se va a ofrecer una visión general de los temas que se encuentran relacionados con el desarrollo del proyecto de fin de carrera.

Los temas que se van a tratar son:

- Desarrollo de aplicaciones en dispositivos móviles.
- Arquitectura cliente-servidor.
- Protocolos de comunicación en dispositivos móviles.
- Métodos empleados en las conexiones seguras.
- Descripción del sistema de gestión de información turística en el que se engloba el proyecto.

2.1.1. Desarrollo de aplicaciones en dispositivos móviles

Antes de empezar a explicar el estado actual en el que se encuentra el desarrollo de aplicaciones en estos terminales, vamos a dar una definición de lo que son los dispositivos móviles: es un pequeño aparato con ciertas capacidades de procesamiento, con conexión permanente o intermitente a una red, con memoria limitada, diseñado específicamente para una sola función, aunque puede realizar otras más generales. Ejemplos: un teléfono móvil, una PDA, un Smartphone, etc. *(Definición obtenida de Wikipedia)*

Figura 1: Dispositivos móviles.

Actualmente, el desarrollo de aplicaciones en dispositivos móviles está experimentando un gran auge, debido en gran medida a los avances tecnológicos que se están produciendo en el campo de la electrónica.

A día de hoy existen pocas compañías que desarrollen aplicaciones para estos dispositivos. Las principales compañías son:

- Microsoft Corporation.

  Esta es la compañía más importante de las que vamos a mencionar, es líder mundial en el desarrollo de software, servicios y tecnologías de Internet para sistemas de uso personal y de negocios.

  Microsoft además de ser el líder mundial en el terreno de la informática, posee una gran influencia dentro del mundo del desarrollo de aplicaciones en dispositivos móviles, y tiene varias versiones de sistemas operativos, aplicaciones y plataformas de desarrollo.

  Los sistemas operativos distribuidos son (ordenado del más antiguo al más actual): PocketPc 2002, Windows Mobile 2003, Windows Mobile 2003 Second Edition, Windows Mobile 5.0 y Windows Mobile 6.0.

  Todos estos sistemas operativos tienen la propiedad de ser compactos y de traer una suite de aplicaciones básicas basadas en la API Win32 de Microsoft.

  Los dispositivos que pueden utilizar Windows Mobile son los PocketPC's, los Smartphones y el Media Center portátil.

  Después de ver los sistemas operativos disponibles, tendríamos que indicar cuáles son las aplicaciones más destacables para un dispositivo móvil, pero sin ninguna duda la más importante es “ActiveSync”.

  ActiveSync es una aplicación de sincronización para dispositivos móviles basados en Windows. La aplicación realiza la sincronización entre un dispositivo móvil y un ordenador, permitiendo la transferencia de documentos de trabajo, de imágenes, de música, de vídeos y de aplicaciones desde el dispositivo, pudiendo mantenerlo actualizado.

  La plataforma que aporta la eficacia del entorno de programación .NET Framework a los dispositivos móviles es .Net Compact Framework. Esta plataforma es un entorno independiente del hardware para la implementación de programas en dispositivos con limitaciones computacionales, como son PDA’s (asistentes de datos personales), Pocket PC’s, teléfonos móviles, descodificadores de televisión, dispositivos de computación para automóviles (dispositivo GPS), etc.

  .Net Compact Framework es un subconjunto de la biblioteca de clases .NET Framework, aunque también incluye clases diseñadas

expresamente para él. Otra característica muy importante es que hereda la arquitectura .NET Framework completa de Common Language Runtime (CLR) y la ejecución de código administrado.

Para poder desarrollar software sobre esta plataforma necesitaremos utilizar alguna de estas dos herramientas: Microsoft Embedded Visual Tools o Microsoft Visual Studio. Ambos entornos de desarrollo llevan integrado un emulador de dispositivos móviles para poder realizar las diferentes pruebas de nuestras aplicaciones sin la necesidad de poseer un dispositivo real.

- PalmSource, Inc.

  Esta empresa es una de las dos que se crearon tras la división de Palm, Inc. en el 2003. Tras la separación la empresa se dedicó al desarrollo del sistema operativo, Palm OS, para dispositivos móviles.

  PalmSource, Inc. es junto a Microsoft, la única compañía que trabaja en el desarrollo de sistemas operativos para dispositivos móviles, aunque la mayor parte de sistemas operativos implantados en los dispositivos móviles pertenecen a Microsoft.

  En la actualidad, está trabajando en el desarrollo de nuevas versiones del sistema operativo, una de la cuales funcionará sobre un núcleo de LINUX.

- Electronic Arts (EA).

  Electronic Arts es una empresa norteamericana de software de entretenimiento multiplataforma, es decir, desarrollan videojuegos tanto para ordenadores, como para videoconsolas, teniendo en cuenta también los dispositivos móviles (en mayor medida para teléfonos móviles).

  Algunos ejemplos de software para dispositivos móviles que desarrolla esta empresa podrían ser los juegos: FIFA 07, Need for Speed Carbono, Los Sims,…, claro está, todo en materia de entretenimiento.

  Cabría destacar el gran esfuerzo que está realizando para el desarrollo de estas aplicaciones en estos terminales.

Con lo expuesto anteriormente nos podemos hacer una idea del gran progreso que se está produciendo en este campo de los dispositivos móviles, debido a la implicación de grandes compañías y de los avances en los terrenos de la tecnología y de la informática.

2.1.2. Arquitectura cliente-servidor

La arquitectura cliente-servidor consiste básicamente en que una aplicación, el cliente, realiza peticiones a otra aplicación, el servidor, que le da respuesta.

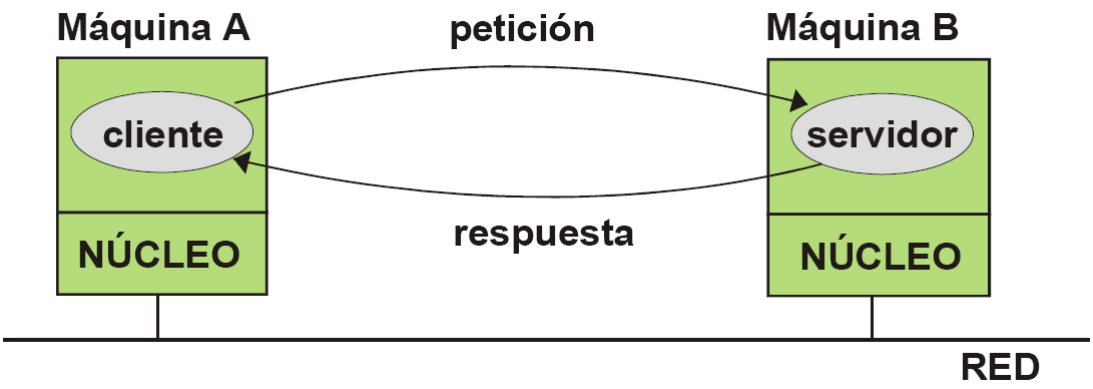


Figura 2: Arquitectura Cliente-Servidor.

Esta arquitectura fue diseñada para utilizarse en sistemas operativos multiusuario distribuidos a través de una red de ordenadores, sin embargo también puede ser aplicada a aplicaciones que sean procesadas en un solo equipo, aunque no sea posible obtener tanto rendimiento.

Diseñar una aplicación con esta arquitectura, nos da la capacidad de separar responsabilidades (funcionalidades) y gestionar la información de forma centralizada, ya que el procesamiento está repartido entre los clientes y los servidores. Esto contribuye a la realización de un diseño más claro de la aplicación.

También tendríamos que destacar la característica de desarrollar aplicaciones escalables, es decir, seríamos capaces de aumentar las funcionalidades tanto de los clientes como de los servidores de forma independiente.

En esta arquitectura, la separación que existe entre el cliente y el servidor es de tipo lógica, ya que el servidor no se ejecuta obligatoriamente sobre un solo equipo ni tampoco sobre un solo programa.

Esta última característica expuesta da lugar a la creación de sistemas multicapa, el servidor está formado por distintos programas que pueden ser ejecutados en diferentes ordenadores; aumentado de esta forma el grado de distribución de la aplicación o sistema.

El funcionamiento de la arquitectura cliente-servidor es muy simple. El servidor espera de forma pasiva las peticiones de los clientes, para procesarlas y responder al cliente con la ejecución de su solicitud. Por su parte, los clientes envían las peticiones al servidor y esperan hasta que el servidor les manda el resultado de las mismas.

También podemos señalar dentro del funcionamiento, que los servidores pueden ser de dos tipos: sin estado o con estado. Los servidores sin estado no pueden guardar ninguna información entre las peticiones, un ejemplo son los servidores de HTTP (HyperText Transfer Protocol). En cambio, los servidores con estado pueden almacenar la información entre las distintas peticiones, siendo a nivel de aplicación o de sesión.

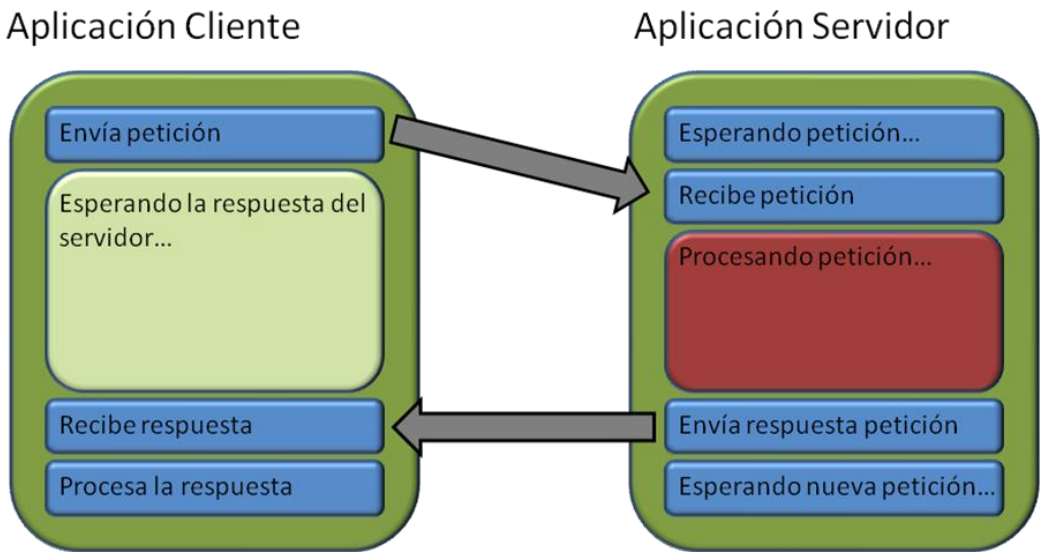


Figura 3: Diagrama de funcionamiento Cliente-Servidor.

Además, debemos decir que la arquitectura cliente-servidor ha influido en gran medida en otra arquitectura de red, como es la arquitectura peer-to-peer (en español, par-a-par). Esta arquitectura permite actuar a un nodo o a una aplicación como cliente y servidor a la vez. Un ejemplo de esta arquitectura serían los programas de transferencia de archivos, como Emule, Bittorrent, Ares,…

Para concluir con este apartado podríamos decir que la arquitectura cliente-servidor es la más utilizada en la implementación de sistemas distribuidos. Un buen ejemplo que demuestra esto, sería la visita a una página Web, en la que existe un servidor Web que sirve las páginas al navegador de internet.

2.1.3. Protocolos de comunicación en dispositivos móviles

En la actualidad, existe una gran cantidad de medios de comunicación para transmitir la información necesaria de la forma más cómoda y fácil posible. Los protocolos de comunicación están en continua evolución, e incluso aparecen nuevos protocolos cada poco tiempo, ya que se está trabajando muy duramente en este campo.

Los protocolos de comunicación que están teniendo mayor éxito son los protocolos de comunicación inalámbricos, es decir, aquéllos que no utilizan como medio físico para la transmisión de la información el cable.

Dentro de estos protocolos de comunicación inalámbricos, hay que destacar aquéllos que han experimentado una gran evolución o los nuevos que están apareciendo, debido al avance tecnológico de los dispositivos móviles.

WIFI (Wireless Fidelity). Es un conjunto de estándares para redes inalámbricas basados en la especificación IEEE 802.11. Tiene como finalidad ser utilizado en redes locales inalámbricas, aunque también es frecuente en la actualidad que se utilice para acceder a internet.

Esta tecnología sufre en la actualidad un gran problema de seguridad, ya que en muchas ocasiones se implantan redes sin proteger los datos que circulan por ella. Este problema se soluciona con el uso de protocolos cifrados, como son WEP (Wired Equivalent Privacy) y WAP (Wireless Application Protocol).

Otro problema o desventaja que se debería tener en cuenta es la pérdida de velocidad de conexión, debido a las interferencias y pérdidas de señal que pueda provocar el entorno.

Hay que tener en cuenta que para usar esta tecnología se necesitan una serie de dispositivos que nos permitan la emisión y recepción de la señal.

Para la emisión de la señal, los dispositivos más utilizados son los routers y los puntos de acceso. En el caso contrario, para la recepción de la señal, se utilizan tarjetas (PCI o USB) conectadas en los ordenadores.

GSM (Global System for Mobile communications). Es un estándar mundial para teléfonos móviles digitales, destacando que es un estándar abierto, no propietario y evolutivo (ya que se encuentra todavía en desarrollo).

Una de las principales características del estándar GSM, es que los canales de voz y las señales son digitales, además de poseer un nivel de seguridad aceptable.

Las frecuencias que utilizaba GSM en su principio eran de 900 MHz, pero en la actualidad se usan frecuencias de 1800 y 1900 MHz. Por este motivo, los teléfonos móviles actuales son tribanda.

Otro punto a destacar es que se permite la comunicación tanto de voz como de datos.

*GPRS (General Packet Radio Service).*Es una tecnología digital utilizada en la telefonía móvil.

Se trata de una modificación de la red GSM, de la transmisión de datos, pasando de una conmutación de circuitos (donde el circuito queda reservado de forma permanente mientras la comunicación dure, aunque no exista transferencia de datos) a la conmutación de paquetes.

Esta modificación provoca que se pase de velocidades de 9.6 kbps de GSM a 40 kbps en recepción y 20 kbps de transmisión en GPRS.

Actualmente, esta tecnología está siendo superada por la siguiente versión de GSM, denominada UMTS (Universal Mobile Telecommunications System).

Bluetooth. Es el nombre común de la especificación IEEE 802.15.1, la cual define un estándar global de comunicación inalámbrica. Este estándar da la capacidad de transmitir voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia segura.

Los objetivos fundamentales del estándar son facilitar las comunicaciones entre los terminales móviles y fijos, eliminar conectores y cables entre éstos, y facilitar la creación de pequeñas redes inalámbricas.

El desarrollo de este estándar ha sido posible gracias en gran medida a la participación de grandes compañías, como Sony Ericsson, Nokia, Intel o Motorola, entre otras.

Con el avance tan rápido que está sufriendo, va a da lugar, en un futuro no muy lejano, a una conectividad inalámbrica total tanto en el hogar como en el lugar de trabajo.

Los dispositivos que más monopolizan esta tecnología son los terminales móviles, como PDA's, teléfonos móviles, etc.

IrDA (Infrared Data Association). Define un estándar de nivel físico para la transmisión y recepción de datos por rayos luminosos que se mueven en el espectro infrarrojo. Permite que la comunicación entre dos extremos sea bidireccional, y las velocidades oscilan entre 9.600 bps y los 4 Mbps.

Actualmente se encuentra en estudio una nueva versión, que alcanza velocidades de 16 Mbps (de forma teórica). Su nombre es FIR (Fast Infrared).

Esta tecnología se encuentra muy frecuentemente en dispositivos móviles (sobre todo en teléfonos móviles) y ordenadores portátiles.

Aunque, la inmensa mayoría de los dispositivos móviles, como es lógico, utilizan protocolos de comunicación inalámbricos, tenemos que decir que uno de los medios de comunicación más utilizados para la comunicación de datos en los dispositivos móviles, es el empleo de la conexión USB (Universal Serial Bus).

La conexión USB se realiza mediante la utilización de un cable que conecta el dispositivo con otro terminal (generalmente un ordenador), permitiendo la transferencia de información sin ningún tipo de problema.

2.1.4. Métodos empleados en las conexiones seguras

Las conexiones seguras surgen debido al gran tráfico de información confidencial que existe en la red, como pueden ser las transacciones realizadas por los clientes de un banco, las compras efectuadas en una tienda online, etc.

Todas las comunicaciones que se realizan sobre conexiones seguras utilizan algoritmos de cifrado tanto simétrico como asimétrico, para cifrar la información que circula por sus redes.

Para poder entender los algoritmos de cifrado de una manera más fácil son necesarias una serie de nociones básicas relacionadas con la criptografía.

La criptografía es una disciplina que estudia los principios, métodos y medios para transformar los datos y así ocultar la información que contienen.

Para llevar a cabo este ocultamiento de la información, haremos uso de un criptosistema o sistema de cifrado. A continuación, se muestra el diagrama de un criptosistema.

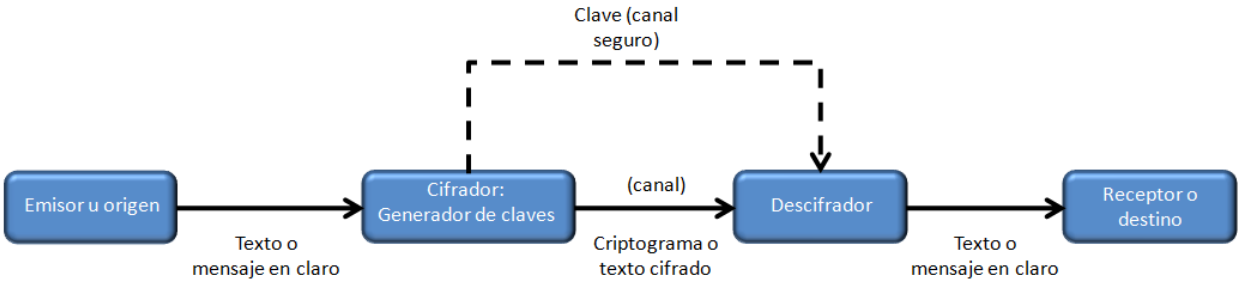


Figura 4: Diagrama de un criptosistema.

El cifrador será un dispositivo o un programa que implementará un algoritmo de cifrado: función matemática o clave de cifrado.

Al sistema de cifrado se le podría añadir un protocolo de intercambio o de negociación de clave: procedimiento en el que dos sistemas de cifrado interconectados cambian información a fin de determinar qué claves usarán en el cifrado del tráfico entre ellos.

Los criptosistemas o sistemas de cifrado se clasifican según varios criterios:

- Naturaleza:
    - Sustitución: consiste en sustituir unos símbolos por otros mediante el desplazamiento del alfabeto del mensaje en un número determinado de unidades.
    - Transposición o permutación: este algoritmo altera el orden de los símbolos del mensaje mediante el empleo de patrones geométricos, permutaciones de grupos o posicionamiento en zig-zag.
    - Producto: consiste en aplicar sucesivos algoritmos de cifrado a partir del mensaje en claro. Cada uno de los algoritmos actúa sobre el mensaje cifrado obtenido por el algoritmo anterior.
- Clave empleada:
    - Simétricos: el cifrado y el descifrado se realiza utilizando la misma clave.
    - Asimétricos: algoritmo basado en el empleo de un par de claves, una privada (descifrado) y otra clave pública (cifrado), para el cifrado y descifrado de los mensajes.
- Número de símbolos a cifrar:
    - Flujo: consiste en descomponer el mensaje en símbolos y cifrar cada uno con el correspondiente símbolo de una clave de cierta longitud.
    - Bloque: fragmenta el mensaje en bloques de símbolos de igual longitud que la clave. Todos los bloques son cifrados con la misma clave.

De todos estos criterios los que más nos interesan son los de clave simétrica y asimétrica.

2.1.4.1. *Sistema de cifrado simétrico*

Tiene como principal característica el uso de una única clave para cifrar y para descifrar la información de los mensajes.

Esto provoca que tanto el emisor como el receptor tengan que ponerse de acuerdo previamente sobre la clave que van a utilizar, ya que una vez que ambos tengan la clave, el emisor podrá cifrar el mensaje usando dicha clave y el receptor lo podrá descifrar con la misma, una vez haya recibido el mensaje.

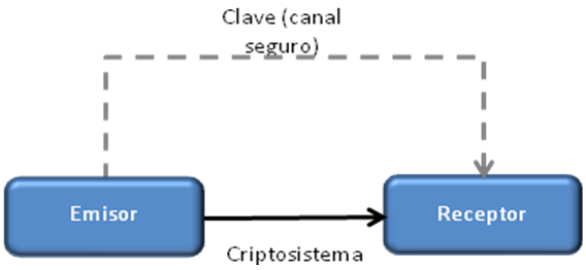


Figura 5: Esquema de cifrado simétrico.

En estos cifrados toda la seguridad recae en la clave, y no en los algoritmos; es muy importante que el espacio de posibilidades de claves sea lo más amplio posible. La ampliación del espacio de posibilidades de claves se realiza mediante el incremento de la longitud de las claves.

Actualmente se utilizan claves con una longitud de 128 bits, dando lugar a 2^{128} claves posibles, porque un buen criptosistema pone toda la seguridad en la clave y ninguna en el algoritmo, es decir, no debería ser de ninguna ayuda para un atacante conocer el algoritmo de cifrado que se está usando, sino conocer la clave. Por ejemplo, los sistemas GNU poseen algoritmos con esta propiedad.

Este tipo de cifrados poseen varios inconvenientes. El principal de estos problemas es el intercambio de las claves, debido a que se necesita un canal seguro en el que realizar el intercambio, para que un atacante no intercepte la clave, ya que para él sería más fácil intentar obtener la clave antes de probar todas las posibles. Otro inconveniente, sería el altísimo número claves que necesitaría el criptosistema si fuera utilizado por muchos usuarios.

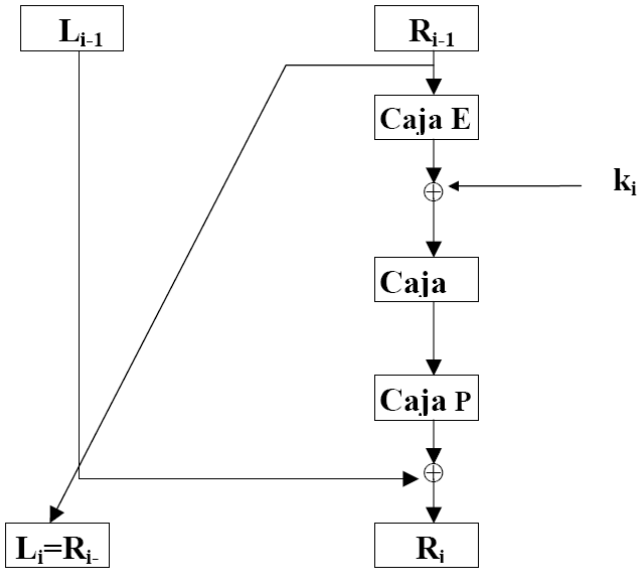
Número de claves en el criptosistema = $N \cdot (N - 1) / 2$; donde N es el número de usuarios, y donde cada usuario tendría que almacenar N – 1 claves.

Un ejemplo de sistemas de cifrado simétrico sería la máquina Enigma, empleada por los alemanes durante la Segunda Guerra Mundial, en la que las claves se difundían en libros de códigos.

Actualmente, los criptosistemas simétricos más utilizados son IDEA (International Data Encryption Algorithm), DES (Data Encryption Standard) y Rijndael o AES (Advanced Encryption Standard). Los dos últimos los explicaremos a continuación.

Sistema de cifrado simétrico DES (Data Encryption Standard). Es un algoritmo de cifrado por bloques, con un tamaño de 64 bits. La longitud de la clave es de 64 bits, pero en realidad solo utiliza 56 bits para el cifrado del mensaje, ya que los 8 bits restantes se utilizan en la comprobación de la paridad, es decir, la longitud de la clave real es de 56 bits, contribuyendo a un corto espacio de posibles claves.

Proceso de cifrado



Proceso de generación de claves

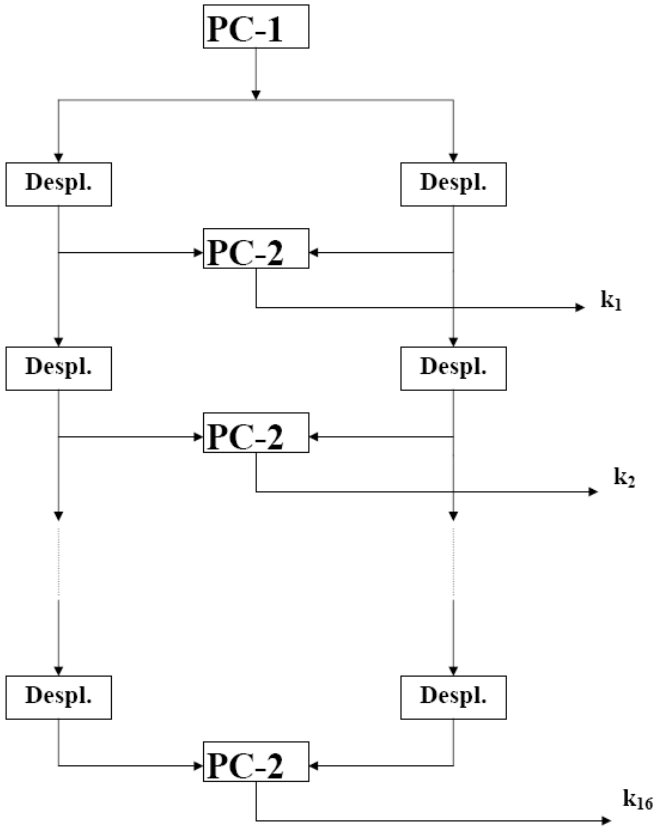


Figura 6: Esquema del algoritmo DES.

Actualmente, el algoritmo DES se considera no fiable para muchos sistemas e incluso ha sido sustituido como cifrador estándar por el algoritmo AES, debido principalmente a la longitud de su clave, ya que en menos de un día es posible averiguarlas.

Por este motivo, han aparecido nuevas variantes del algoritmo como son el doble DES, el triple DES con dos claves, triple DES con tres claves..., este último es el más seguro y la más utilizada de las variantes.

Sistema de cifrado simétrico Rijndael o AES (Advanced Encryption Standard). El algoritmo fue desarrollado por los criptólogos Vincent Rijmen y Joan Daemen.

El sistema consiste en una red de sustitución-permutación, que posee un esquema de cifrado de bloques, con un tamaño 128 bytes o de múltiplo de 4 bytes. La longitud de la clave es variable, pudiendo ser de 128, 192 y 256 (estándar) bits o bien múltiplos de 4 bytes.

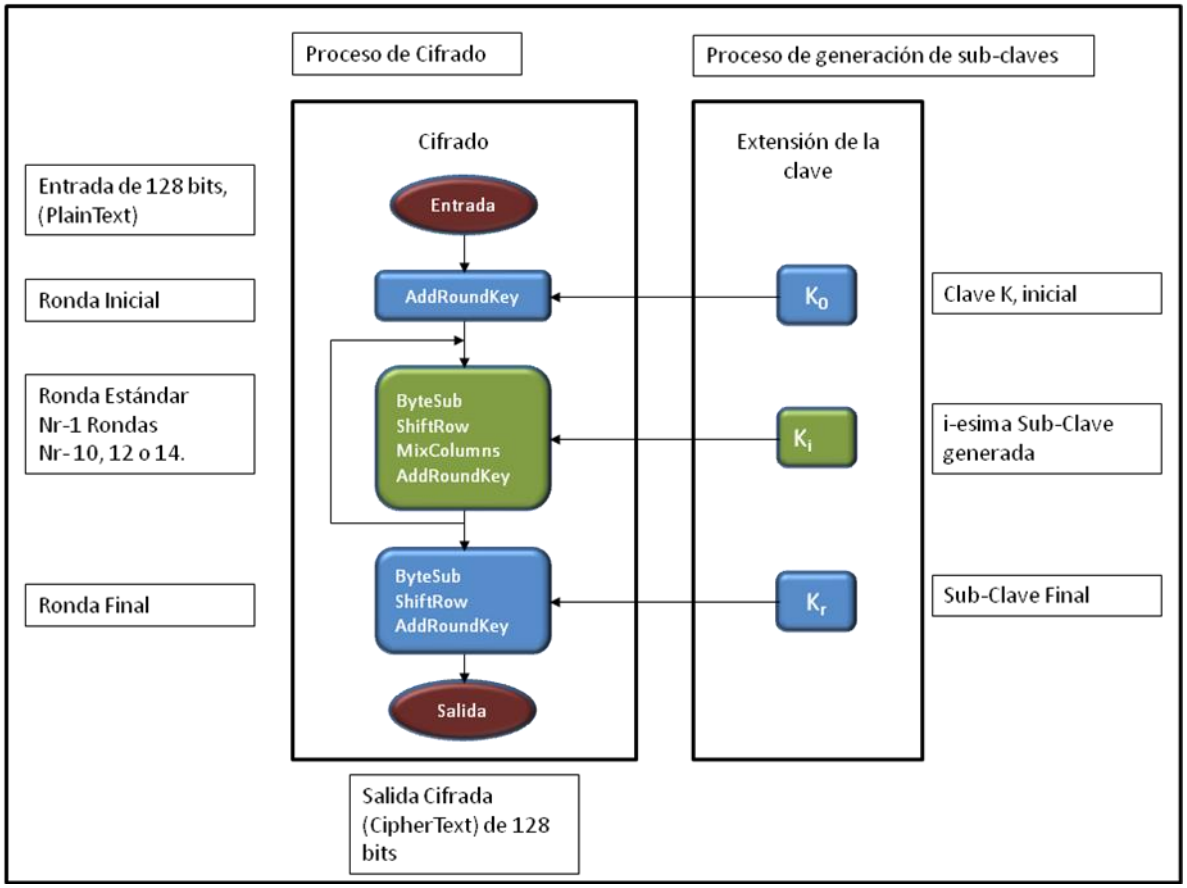


Figura 7: Esquema general del algoritmo AES.

El algoritmo AES es muy rápido en ejecución, tanto en software como en hardware, y requiere poca memoria. Además, es relativamente fácil de implementar.

Desde que se convirtió en un estándar de cifrado es uno de los algoritmos más utilizados en la criptografía simétrica, sustituyendo a su predecesor el algoritmo DES.

2.1.4.2. *Sistema de cifrado asimétrico*

Tiene como principal propiedad el uso de un par de claves, por usuario, para el envío de datos de forma cifrada.

El par de claves de un usuario está formado por una clave pública que puede conocer cualquier persona, que se utiliza para cifrar el mensaje por parte del emisor, y una clave privada que sólo conocerá el receptor (usuario al que pertenece el par de claves), con la que descifrará el contenido del mensaje. La clave pública y privada de un usuario son diferentes.

Figura 8: Esquema de cifrado asimétrico.

Los sistemas de cifrado asimétrico tienen como principales ventajas:

- No exige un canal seguro para distribuir la clave pública.
- El número de claves empleadas es sólo dos veces el número de usuarios.
- Cada usuario almacenará N+1 claves, siendo N el número de usuarios en el sistema. (N-1 públicas del resto de usuarios, más su clave pública y clave privada).

Estos sistemas tienen un pequeño problema, ya que no permiten la autenticación del emisor, es decir, no sabríamos quién es el emisor de un mensaje, debido a que la clave pública puede ser conocida por cualquiera. Este inconveniente puede ser solucionado con el empleo de la firma digital.

Algunos de los principales algoritmos y protocolos que utilizan este tipo de cifrado son Diffie-Hellman, ElGamal, RSA, SSL (Secure Socket Layer), TLS (Transport Layer Security), etc.

Sistema de cifrado de clave pública RSA. Es un algoritmo asimétrico cifrador de bloques. Utiliza una clave pública, la cual se distribuye de forma autenticada (preferiblemente), y otra privada, que guarda el propietario en secreto.

Si queremos usar este algoritmo en el cifrado de un mensaje, lo único necesario que tendría que realizar el emisor sería buscar la clave pública del receptor, cifrar el mensaje con esta clave y enviar el mensaje cifrado al receptor. Posteriormente, el receptor se encargará de descifrar el mensaje mediante el uso de su clave privada.

Tendríamos que destacar que la seguridad del algoritmo radica en las claves, ya que estas se generan mediante el producto de dos números primos muy grandes (mayores a 10^{100}) elegidos al azar y el empleo de expresiones exponenciales en aritmética modular.

Por estos motivos, el cifrado de clave pública RSA es de los más utilizados cuando no existe un canal de comunicación seguro para el empleo de un sistema de cifrado simétrico. Aunque, actualmente también se utiliza de forma combinada con sistemas de cifrado simétrico. Un ejemplo sería el empleo combinado del sistema de cifrado TripleDES con el sistema RSA para el cifrado de sesiones, en la que las conexiones son cifradas con el TripleDES y la clave de éste es cifrada con RSA.

2.1.5. Descripción del sistema de gestión de información turística

El sistema en el que se va a desplegar el módulo de comunicaciones es un sistema encargado de gestionar información de interés turístico, es decir, es capaz de ofrecer a los clientes mapas de una zona turística, un lista de monumentos o lugares de interés de dicha zona e incluso realizar planificaciones de rutas.

Debido a la alta complejidad del sistema, se encuentra dividido en varios módulos, los cuales son:

- Aplicación de usuario en el dispositivo móvil (PDA). Aplicación desarrollada en un dispositivo móvil para satisfacer las peticiones turísticas, incluidas en el sistema, que un usuario quiere realizar. Contiene la interfaz gráfica de usuario (en inglés Graphical User Interface, GUI).

- Módulo de comunicación, estará desplegado en el dispositivo móvil y en el servidor del sistema. Es el encargado del establecimiento de la comunicación entre la aplicación del dispositivo móvil y los servidores que satisfarán las peticiones solicitadas. Desarrollado en este PFC.

- Módulo de gestión de mapas. La función principal del módulo es obtener los mapas de una zona determinada, a través de internet, y enviárselos a la aplicación del usuario.

- Módulo de gestión de monumentos. Módulo que tiene como principal objetivo el tratamiento de toda la información relativa a los monumentos y lugares de interés turístico.

- Planificador de rutas. Este módulo tiene la capacidad de procesar una lista de monumentos, y devolver al usuario una ruta completa (horarios, transportes a utilizar, duración de los trayectos,…) en la que ver los monumentos solicitados. Para ello utiliza un planificador.

- Módulo de gestión de transportes. El módulo tiene como finalidad calcular el tiempo que se tarda en llegar de unos monumentos o sitios de interés a otros utilizando diferentes medios de transporte.

Estos cuatro últimos módulos explicados, módulo de gestión de mapas, módulo de gestión de monumentos, módulo de gestión de transportes y planificador de rutas, se encontrarían desplegados en el servidor del sistema, al igual que un bloque del módulo de comunicación, ya que el otro bloque formará parte de la aplicación implementada en el dispositivo móvil.

Para entender más fácilmente el modelado y el funcionamiento del sistema en el que se va a desarrollar el proyecto (módulo de comunicación), se muestra a continuación un diagrama del sistema y la explicación de su funcionamiento.

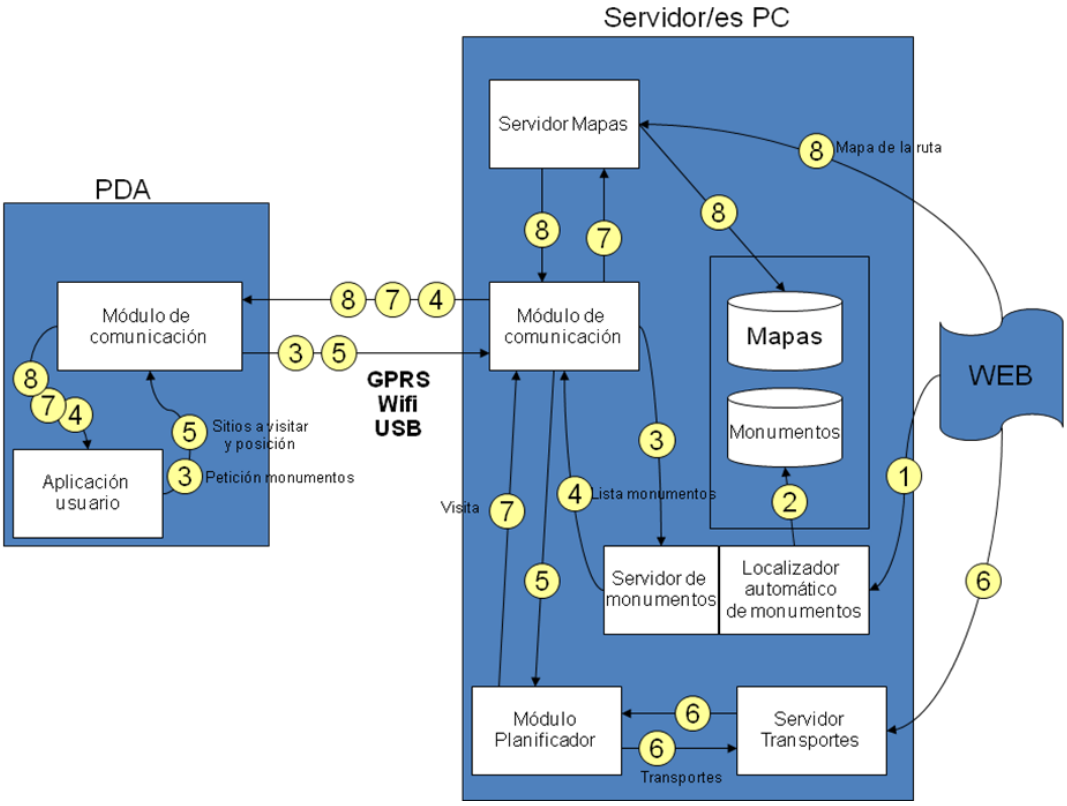


Figura 9: Diagrama del sistema.

Explicación del funcionamiento, por pasos:

1. y 2. El módulo de monumentos se encarga de rellenar la base de datos con los monumentos, horarios, direcciones..., que ha obtenido de internet o de los puntos de interés de TomTom.

3. La aplicación de la PDA le pide al módulo de monumentos que le envíe el fichero con los monumentos que hay disponibles.

4. El usuario selecciona los monumentos asignando preferencias entre ellos si lo considera oportuno y envía la lista de los que le gustaría visitar junto con su posición actual, determinada por el GPS de la PDA.

5. La lista de monumentos llega al planificador.

6. Éste llama al servidor de transportes que calcula lo que se tarda en llegar de unos monumentos o sitios de interés a otros utilizando diversos medios de transporte. Con esta información se crea un problema de planificación en PDDL (Planning Domain Definition Language).

7. El planificador encuentra un plan para visitar los monumentos y envía la propuesta de la visita a la PDA y al módulo de mapas, para que se obtenga un mapa en el que aparezcan todos.

8. El módulo de mapas recoge el mapa de internet y lo envía a la PDA, mediante el uso del módulo de comunicación.

9. El usuario acepta la propuesta o realiza modificaciones volviendo en el segundo caso al punto cuatro.

Este sería el funcionamiento inicial, luego la PDA irá enviando información sobre la situación del usuario y el grado de cumplimiento del plan para replanificar si ocurre algo, o pedir mapas más detallados de diversas zonas...

Después de dar la descripción y el funcionamiento del sistema en el que se va a desplegar el módulo de comunicación que se va a desarrollar en el proyecto, en el siguiente capítulo procederemos a realizar la descripción del módulo de comunicación e indicar cuáles son los objetivos generales de éste.

2.2. Tecnologías utilizadas

En primer lugar tendremos que elegir la tecnología utilizada para la plataforma de desarrollo, que en este caso es la plataforma .Net de Microsoft. Esta plataforma de desarrollo es gratuita, es decir, no necesita ninguna licencia para su utilización.

Se ha escogido esta plataforma porque el dispositivo móvil (PDA modelo HP 6515w) y el equipo donde van a ser desplegados los módulos del PFC utilizan versiones del sistema operativo Microsoft Windows. En concreto, la PDA utiliza Microsoft Windows Mobile 2003 y el equipo del servidor Microsoft Windows Vista.

La plataforma .Net es una infraestructura sobre la que se constituye todo un conjunto de lenguajes y servicios que simplifican enormemente el desarrollo de aplicaciones. Los componentes principales de este entorno de desarrollo son:

- Lenguajes de compilación
- Biblioteca de clases de .Net
- Common Language Runtime (CLR)

Una característica importante de esta plataforma, es el soporte de múltiples lenguajes de programación, aunque cada lenguaje tenga sus propias características, teniendo en cuenta, que la herramienta compila el código fuente de cualquiera de los lenguajes soportados en un mismo código, denominado código intermedio (MSIL, Microsoft Intermediate Lenguaje). Esto hace posible desarrollar cualquier tipo de aplicación con cualquiera de estos lenguajes.

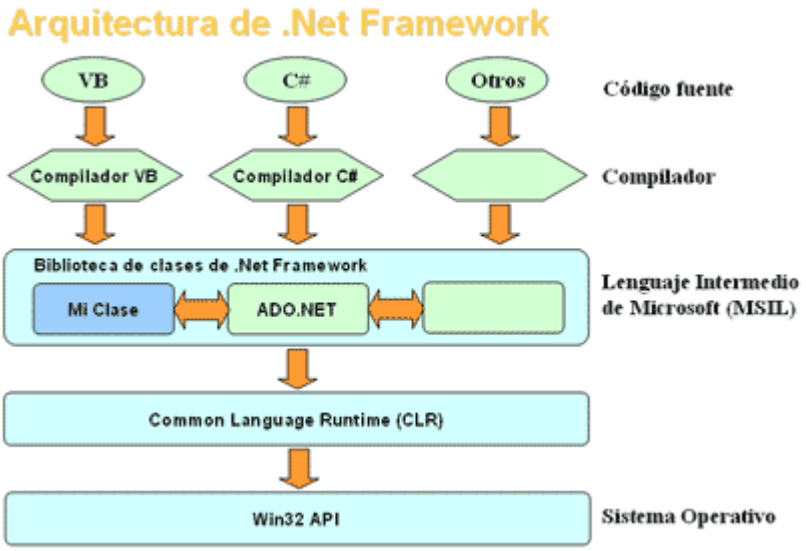


Figura 10: Arquitectura de .Net Framework.

Después de seleccionar la plataforma de desarrollo deberemos elegir uno de los lenguajes de programación que posee .Net Framework para realizar la implementación del módulo de comunicaciones del sistema. El lenguaje de programación empleado para la implementación es Visual Basic .Net. El motivo por el que se eligió este lenguaje de programación fue fundamentalmente por su fácil aprendizaje para nuevos programadores.

El lenguaje de programación Visual Basic .Net es un lenguaje orientado a objetos, al igual que el resto de lenguajes de la plataforma, esto permite hacer las aplicaciones y módulos más fáciles de escribir, mantener y reutilizar.

Aunque no es el único lenguaje de programación utilizado en la implementación de este PFC, ya que en la realización de la parte web de la aplicación se han utilizado:

- HTML (HyperText Markup Language): aunque no lo podemos definir propiamente como un lenguaje de programación sino más bien como un lenguaje de marcas de texto utilizadas para la realización de páginas Web. Lo utilizaremos porque es el formato estándar de las páginas web.

- VBScript (Visual Basic Script): es un subconjunto de lo que se denomina Visual Basic for Application (VBA) que, a su vez es un subconjunto del lenguaje Visual Basic. VBScript fundamentalmente se emplea para integrarse dentro de páginas HTML para ampliar las características de un sitio Web. La preferencia de utilizar este lenguaje antes que otro de los existentes para la creación de páginas ASP es por ser un subconjunto de Visual Basic.

- JavaScript (JScript): es un lenguaje de programación que no requiere compilación, utilizado principalmente en páginas web. Utiliza una sintaxis semejante a la del lenguaje Java, pero al contrario que este, JavaScript no es un lenguaje orientado a objetos propiamente dicho, ya que no dispone de Herencia, sino que es un lenguaje basado en prototipos. El motivo de utilizar este lenguaje no es otro que la necesidad de realización de pequeñas funciones de validación de los formularios de las páginas web.

- CSS (Cascading Style Sheets): no se puede decir que sea un lenguaje de programación, sino un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML. Emplearemos este lenguaje para la personalización de las páginas web debido a su simplicidad.

También tendríamos que destacar que la parte Web de la aplicación (la parte de administración del módulo de comunicación) se ha desarrollado utilizando la tecnología Microsoft Active Server Page (ASP).

Microsoft Active Server Page (ASP) es una tecnología de script que se ejecuta del lado de servidor y puede ser usado para crear aplicaciones Web dinámicas e interactivas. Una página ASP es una página HTML que contiene scripts que son procesados por un servidor Web antes de ser enviados al navegador del cliente.

La última tecnología empleada en el módulo de comunicación es la relacionada con la base de datos en la que utilizamos Microsoft SQL Server 2005 como gestor de base de datos relacionales.

Microsoft SQL Server 2005 es un sistema de gestión de bases de datos relacionales basada en el lenguaje Transact-SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea.

Una vez seleccionadas las tecnologías que usaremos para el desarrollo del módulo de comunicación, tendremos que elegir cuales serán las herramientas empleadas en el desarrollo. En este caso, las herramientas de desarrollo utilizadas son Visual Studio 2005, Internet Information Services y Microsoft SQL Server 2005. En concreto Internet Information Services lo utilizaremos para desplegar el bloque de administración del módulo de comunicación, implementado en el sistema.

Visual Studio es una herramienta de desarrollo de software de Microsoft, naturalmente orientada hacia su entorno de programación: .NET Framework. Al tratarse de un entorno de desarrollo integrado, aúna todas las herramientas del SDK: compiladores, editores, ayuda, etc., facilitando en gran medida la creación de programas.

Internet Information Services es una herramienta que nos proporciona compatibilidad con servidores Web y FTP (File Transfer Protocol), además de sitios web ASP .Net, contenido dinámico con CGI (Common Gateway Interface) y ASP clásico, y administración local y remota.

CAPÍTULO 3.
OBJETIVOS DEL PROYECTO DE FIN DE CARRERA

El objetivo principal de este PFC es la creación de un módulo de comunicaciones para un sistema de gestión turística, sistema ya presentado en el capitulo anterior. En concreto, el módulo debe **ser capaz de permitir una comunicación entre todos los módulos que componen el sistema**, pero principalmente entre la aplicación de usuario, que reside en el dispositivo móvil, y los módulos que reciben las peticiones del cliente y responde a ellas.

Este módulo de comunicación constará de dos partes diferenciadas, una que se ejecutará sobre un servidor PC y otra que se instalará en una PDA pocket PC. Por lo tanto el objetivo anterior se puede dividir en dos:

Implementación del Módulo de comunicación desplegado en el dispositivo móvil. Este bloque se encargará de transmitir las peticiones realizadas por la aplicación del usuario al servidor del sistema, más concretamente al otro bloque del módulo de comunicación. Esta transmisión de datos se producirá bajo una conexión segura y utilizando diversos protocolos de comunicación.

Implementación del Módulo de comunicación desplegado en el servidor del sistema. Este módulo se encargará de recibir todas las peticiones realizadas por el otro bloque del módulo de comunicación, procesar dichas peticiones para que sean satisfechas por el resto de módulos del sistema, y responder al usuario con la respuesta obtenida de los servidores. Además, debe de realizar la gestión y la administración de los recursos del sistema.

En capítulo 4, se explicará de forma más detallada este funcionamiento.

También cabría destacar otros objetivos, aunque no parezcan tan importantes, como son:

- La comunicación entre la aplicación de usuario y los módulos que reciben las peticiones debe ser fluida, es decir, no debe enviarse información banal al módulo que la vaya a procesar, ya que estaríamos malgastando ancho de banda.

- La comunicación entre la aplicación de usuario y el módulo de comunicación de la parte del servidor del sistema debe de ser una conexión segura, debido a que se intercambia información confidencial del cliente, como el identificador del cliente, el identificador del dispositivo móvil y la contraseña del cliente.

- Tolerar el crecimiento continuo de trabajo de la manera más fluida posible, es decir, el módulo de comunicación debe ser escalable.

- El módulo debe ser capaz de tolerar entradas de información erróneas o la carga excesiva de trabajo sin que deje de funcionar correctamente.

El módulo de comunicación es una parte fundamental del sistema, ya que permite la comunicación entre los módulos restantes. Este módulo está formado a su vez por dos sub-módulos, uno desplegado en la aplicación del dispositivo móvil y el otro en el servidor del sistema.

A continuación, se muestra un diagrama sobre la estructura que poseerá este módulo de comunicación, que hemos desarrollado en el proyecto.

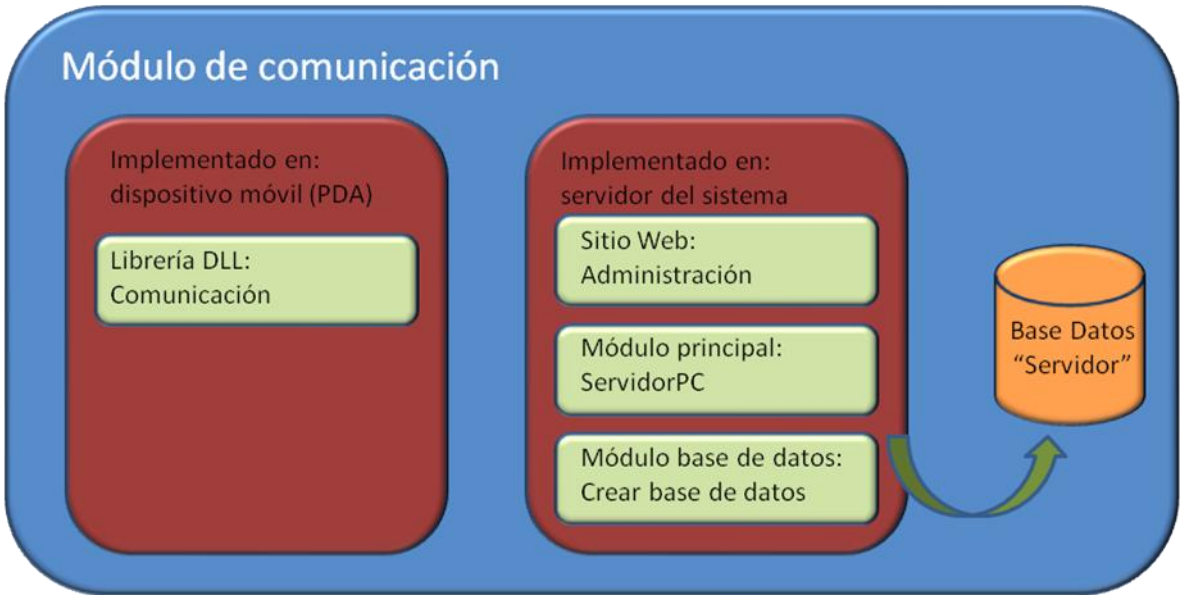


Figura 11: Diagrama del módulo de comunicación.

CAPÍTULO 4.
MEMORIA DEL TRABAJO REALIZADO

4.1. Introducción

En este capítulo de la memoria, comentaremos todo el trabajo que se ha realizado para llevar a cabo el desarrollo del módulo de comunicación.

En el primer apartado del capítulo expondremos la arquitectura del módulo a desarrollar, y detallaremos sus evoluciones hasta llegar a la que finalmente se ha implementado. Para facilitar su entendimiento utilizaremos una serie de gráficos en los que se mostrarán las distintas arquitecturas diseñadas.

En el siguiente apartado se explicará cuales han sido los procedimientos a seguir durante el desarrollo del módulo de comunicación, en el que destacaremos la realización de los diagramas (diagrama de clases, modelo de la base de datos,…) previos a la implementación, la descripción de los principales algoritmos empleados, etc.

4.2. Arquitectura del proyecto

La arquitectura del módulo ha sufrido una serie de modificaciones a lo largo de su desarrollo. Por este motivo, empezaremos a explicar la arquitectura del primer diseño, y posteriormente indicaremos cuáles han sido sus modificaciones hasta llegar a la versión que se ha desarrollado, y cuál ha sido el motivo de dichas modificaciones.

A continuación, procedemos a describir la primera arquitectura que fue diseña para el módulo de comunicación.

El módulo de comunicación está formado por dos sub-módulos, uno desplegado en la aplicación del dispositivo móvil y el otro en el servidor del sistema.

- **Módulo de comunicación de PDA.**

  Este módulo debe ser capaz de recibir los mensajes de las distintas peticiones que realiza la aplicación usuario, es decir, sirve de enlace entre la aplicación del usuario en la PDA y el resto de módulos del sistema. En la versión actual las peticiones pueden ser "Petición de lista de monumentos", "Sitios a visitar", "Posición" y "Mapa de la zona".

Estos mensajes no necesitan ningún tratamiento, es decir, lo único que realizará este modulo será recibir y enviar dichos mensajes al módulo de comunicaciones que se encuentra en Servidor/es PC.

El envío de mensajes se realizará mediante la utilización de ficheros de texto o ficheros XML (eXtensible Markup Language). El formato de estos ficheros es el siguiente:

- Fichero xml:

  <cod_peticion> </cod_peticion >
  <tipo_comunicacion> </tipo_comunicacion>
  <contenido > </contenido >

- Fichero txt:

  COD_PETICION=
  TIPO_COMUNICACION=
  CONTENIDO =

Finalmente en este primer diseño se decidió el empleo de ficheros XML, debido a que el sistema completo utilizará ficheros de tipo XML para el intercambio de información.

Significado de los campos introducidos en el fichero mensaje:

- COD_PETICION: identifica el tipo de petición que se realiza, la cual utilizará el módulo de comunicaciones del Servidor PC para saber a qué otro módulo de la aplicación global debe ser enviada dicha petición para su correcto tratamiento.

- CONTENIDO: contiene la información exacta sobre la petición realizada por el cliente. Este campo no es relevante para el modulo de comunicación.

- TIPO_COMUNICACION: indica el tipo de comunicación que se debe usar a la hora de conectarnos con el servidor PC para la transferencia de mensajes. El establecimiento de la conexión con el servidor tendría que ser llevado a cabo mediante el medio de comunicación indicado en el mensaje.

  En nuestro caso los medios de comunicación que podrán ser utilizados son: USB, BLUETOOTH, WIFI y GPRS; esto conlleva la comprobación de que en nuestro dispositivo móvil se encuentre disponible el medio de comunicación solicitado y además se encuentre activado.

Este módulo de comunicación posee un pequeño fichero de configuración (fichero de texto), que contiene información relativa al usuario de la aplicación y parámetros de configuración para realizar las conexiones con el servidor/es PC. El formato del fichero es el siguiente:

```
#Información del usuario
IDENTIFICADOR=
CONTRASEÑA=

#Configuración de la conexión
IP_SERVIDOR=
PUERTO=
```

Significado de los distintos campos que se encuentran contenidos en el fichero de configuración.

- Información del usuario: este conjunto de campos es importante, ya que sin el identificador de usuario y contraseña no se podría acceder al módulo de comunicaciones de los servidor/es PC y obtener respuesta a las peticiones realizadas. La contraseña se encontrará cifrada mediante el sistema de cifrado simétrico AES.

- Configuración de la conexión: estos campos especifican la dirección y puerto en el que está funcionando el servidor (módulo de comunicación de Servidor/es PC) que debe gestionar las peticiones solicitadas.

  Esta configuración solo sería modificada si se cambian los valores de los parámetros directamente en el archivo de configuración.

- **Módulo de comunicación de SERVIDOR/ES PC.**

Este módulo de comunicación es el encargado de encaminar los distintos mensajes de petición a los módulos que tienen como función el tratamiento de las peticiones.

Para hacer este encaminamiento es necesario acceder a la etiqueta o campo “cod_peticion” del mensaje, que nos indicará cual es el módulo de la aplicación global a la que hay que enviar el mensaje de la petición.

También es necesario para este encaminamiento conocer la dirección y puerto en la que se encuentran corriendo el resto de módulos/servidores de la aplicación global, para poder enviar las distintas solicitudes. Esta información relativa a los otros

módulos será almacenada en un fichero de configuración, que tendrá el siguiente formato:

```
#Conexión a “Servidor Mapas”
IP_SERVIDOR_MAPS=
PUERTO_MAPS=

#Conexión a “Servidor de Monumentos”
IP_SERVIDOR_MONS=
PUERTO_MONS=

#Conexión a “Módulo Planificador”
IP_SERVIDOR_PLAN=
PUERTO_PLAN=

#Conexión a “Módulo de Transporte”
IP_SERVIDOR_TRANS=
PUERTO_TRANS=

#Conexión a “Módulo xxx”
IP_SERVIDOR_xxx=
PUERTO_xxx=
```

Para que a esta aplicación no pueda acceder cualquier cliente existe una autentificación de usuario para poder utilizar los servicios que se ofrecen en la aplicación. Por este motivo cuando el cliente realiza una petición debe enviar el identificador de usuario y la contraseña para que esta petición se realice. Sabremos que el nombre de usuario y contraseña enviadas son correctas cuando coincidan con las que hay dentro de la bases de datos que poseeremos dentro de este módulo.

En este caso, la base de datos sólo estará formada por una tabla, la tabla USUARIO, en la que se almacenará la información relativa a un usuario, formada por el nombre y apellidos del usuario, correo electrónico, identificador, contraseña e identificador de la PDA.

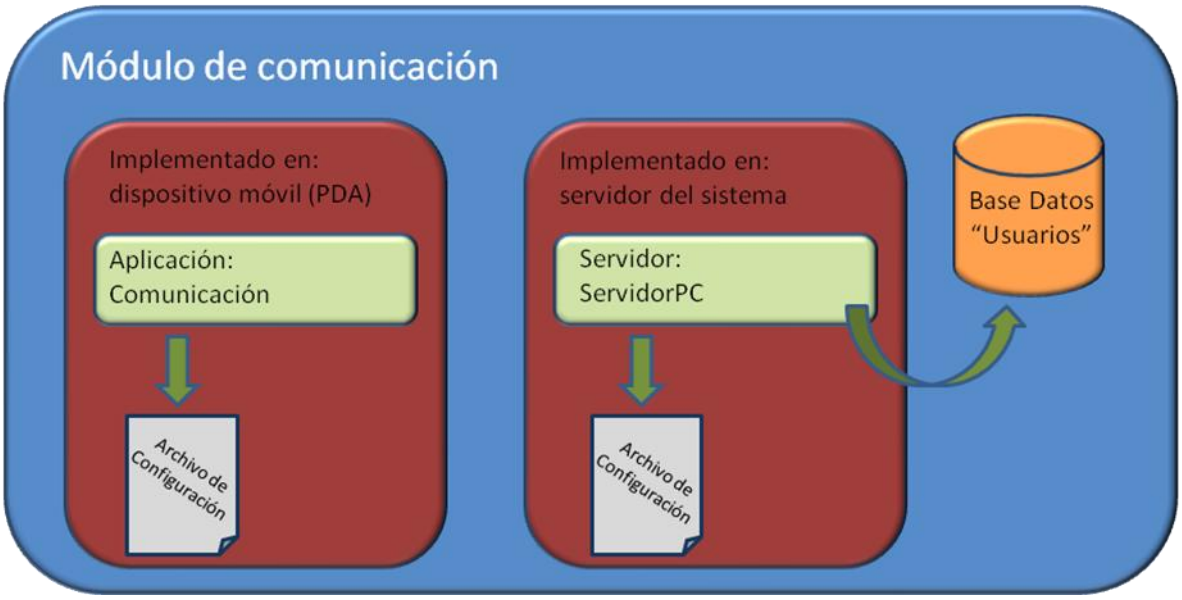


Figura 12: Arquitectura del módulo de comunicación, primer diseño.

Esta es la descripción del primer diseño de la arquitectura del módulo de comunicación, pero finalmente no es la estructura en la que nos hemos basado para la implementación del proyecto. Sin embargo, si ha servido de base para el diseño de la arquitectura final.

Las primeras modificaciones que sufrió esta arquitectura se produjeron en el bloque desplegado en el servidor del sistema, concretamente en el fichero de configuración del servidor.

Este archivo sufrió un cambio en los parámetros de conexión a los diferentes módulos del sistema, para poder realizar una gestión de los módulos (alta y baja) más eficiente, sin necesidad de modificar el código de la aplicación, cada vez que se da de alta o de baja un módulo.

El formato resultante del fichero de configuración es el siguiente:

```
#Conexión a "Modulo_1"
IP_SERVIDOR_ Modulo_1=
PUERTO_ Modulo_1=

#Conexión a "Modulo_2"
IP_SERVIDOR_ Modulo_2=
PUERTO_ Modulo_2=
```

La gestión de los módulos del sistema, tanto el alta como la baja y la modificación, se llevará a cabo directamente modificando el fichero de configuración del bloque desplegado en el servidor del sistema.

La siguiente modificación a la que fue sometido el módulo de comunicación tuvo lugar en ambos bloques del módulo, tanto el desplegado en el dispositivo móvil como el implementado en el servidor. El bloque implementado en el dispositivo móvil pasó de ser una aplicación simple que recibía un fichero XML a convertirse en un pequeño servidor que recibía una petición de la aplicación del usuario. Este servidor trataba de forma secuencial las peticiones que le llegaban, es decir, no ejecutaba una nueva petición hasta que la anterior había sido procesada.

En cambio, el bloque desplegado en el servidor del sistema que procesaba de manera secuencial el tratamiento de las peticiones, pasó a ser un servidor multiproceso, es decir, pasaba a ser un servidor que podía procesar varias peticiones de forma simultánea, sin tener que esperar a que terminará la ejecución de una petición para procesar otra.

Otra modificación que sufrió el diseño del módulo fue en el nivel de seguridad que existía en la comunicación de los dos bloques que pertenecen al módulo de comunicación.

Debido a la transmisión de datos confidenciales por parte del usuario, la comunicación entre módulos debería ser lo más segura posible, aunque el nivel de seguridad en la comunicación de los datos era casi nulo, ya que lo único que se encontraba cifrado era la contraseña del usuario.

Aún así, teníamos un gran problema a la hora de cifrar y descifrar los mensajes, ya que ambos bloques de la aplicación debían de conocer la clave del sistema cifrador, debido a que se empleaba un sistema de cifrado simétrico como es el AES. Por este motivo se decidió utilizar un cifrado simétrico para el contenido del mensaje y un cifrado asimétrico para el cifrado de la clave del sistema simétrico, produciéndose así un aumento bastante significativo en el nivel de seguridad de la comunicación entre ambos bloques. En el apartado siguiente, desarrollo del proyecto, se explicará este proceso con mayor detenimiento.

La siguiente innovación a la que fue sometido es la creación de un fichero de registro de errores producidos en el bloque desplegado en el servidor del sistema.

El formato de salida que posee el archivo de registro de errores (fichero log) es el que se muestra a continuación:

Fecha en la que se produjo el error.
Etiqueta personalizada del error producido.
Tipo de excepción que ha provocado el error en la aplicación.

Un ejemplo del contenido del fichero cuando se produce un error en la aplicación, es el siguiente:

21/08/2007 16:23:22

- ERROR: Interno en el servidor.

- System.Net.Sockets.SocketException: No se puede establecer una conexión ya que el equipo de destino denegó expresamente dicha conexión en System.Net.Sockets.TcpClient.Connect(String hostname, Int32 port) en ServidorPC.Servidor.IniciarSesion() en C:\Users\JavierEnrique\Documents\VisualStudio

Otra de las modificaciones ha sido la actualización de la base de datos, con el objetivo de eliminar los ficheros de configuración y aprovechar la base de datos creada de forma más eficiente.

La actualización de la base de datos se ha realizado con la creación de nuevas tablas: administradores (aquellas personas que tienen privilegios para la modificación de la configuración), módulos (contendrá las configuraciones de acceso, dirección IP y puerto, del resto de módulos del sistema) y configuración (almacena la configuración relativa al servidor).

La siguiente modificación que sufrió el módulo de comunicación se trata de la creación de una nueva aplicación dentro del bloque desplegado en el servidor del sistema, cuyo objetivo es instanciar la base de datos (la base de datos expuesta en el párrafo anterior) que utilizará el módulo de comunicación.

Una de las últimas modificaciones ha sido la creación de un módulo de administración del servidor dentro del bloque desplegado en el servidor del sistema. Este módulo de administración esta implementado como un sitio Web, y tiene como objetivo la gestión de todos los recursos con los que interacciona el módulo de comunicación, como son: la gestión de usuarios, la gestión de módulos, la gestión de administradores del sistema y configuración del servidor. El módulo de administración nos permitirá la gestión de la base de datos utilizada por el servidor del sistema de una forma más cómoda y fácil.

La última modificación para convertirse en la arquitectura que finalmente se ha desarrollado, ha sido la conversión de la aplicación desplegada en el dispositivo móvil por la creación de una librería dinámica (librería DLL, Dynamic Linking Library), que será utilizada por la aplicación de usuario distribuida en los dispositivos móviles.

El principal motivo de la modificación es evitar que el usuario del dispositivo móvil tuviera que ejecutar dos aplicaciones diferentes (con un mayor consumo de procesamiento) para solicitar los servicios ofrecidos por el sistema.

La librería poseerá todos los métodos necesarios para realizar la conexión con el otro bloque del modulo de comunicación, y para la realización de la transmisión de los datos. A consecuencia de esta última modificación se eliminó el fichero de configuración que existía. Ahora la configuración de las conexiones se realiza mediante el uso de parámetros en las llamadas a los métodos y funciones de la librería, aunque existirá una configuración de la conexión por defecto, implementada directamente en el código. En el caso que no sea proporcionada en los parámetros de entrada de los métodos, se utilizará la configuración por defecto.

Con todas estas modificaciones a las que ha sido sometido el módulo de comunicación, llegamos a la arquitectura que finalmente hemos implementado en el proyecto.

A continuación, se muestra la arquitectura que se ha empleado en el desarrollo del módulo de comunicación.

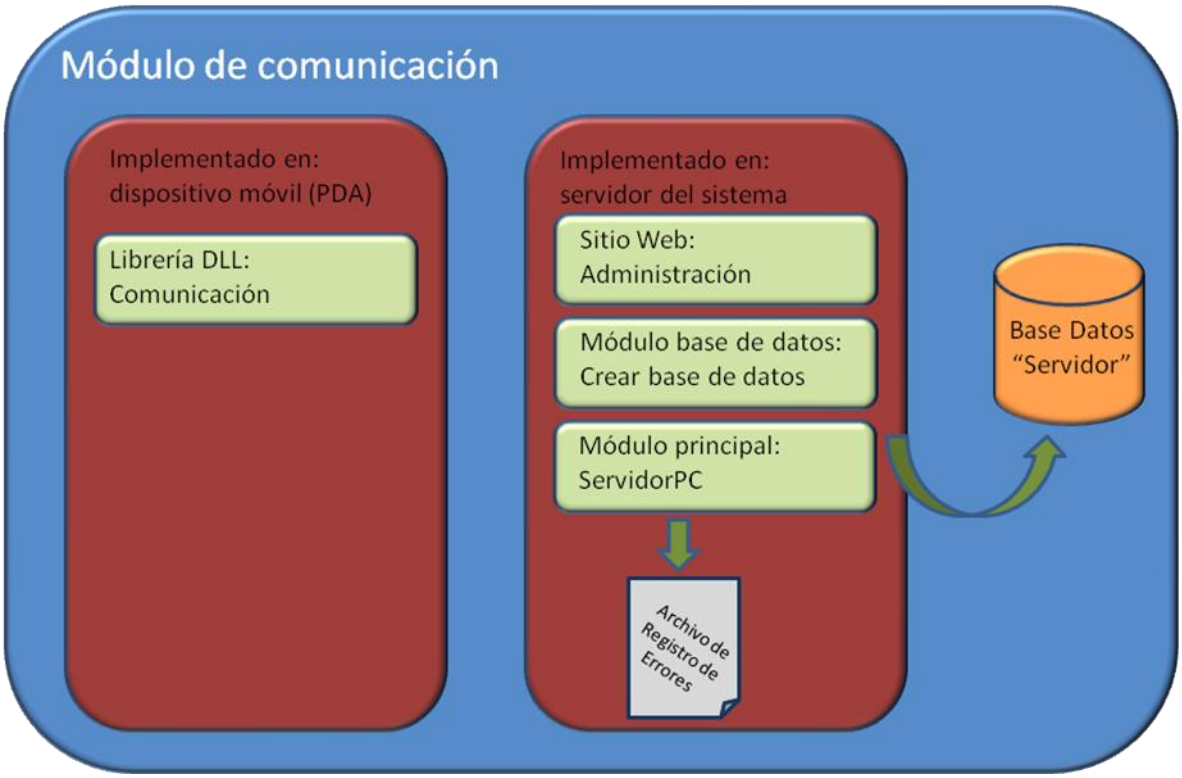


Figura 13: Arquitectura del módulo de comunicación.

La arquitectura se encuentra divida en dos bloques, el módulo implementado en la PDA y el módulo implementado en el servidor del sistema.

El módulo desplegado en el dispositivo móvil está formado sólo por una librería DLL. Este módulo debe ser capaz de transmitir las peticiones realizadas por la aplicación del usuario al servidor del sistema, es decir, sirve de enlace entre la aplicación del usuario en la PDA y el resto de módulos del sistema.

Por otro lado, el módulo implementado en el servidor del sistema debe tener la capacidad de recibir todas las peticiones realizadas por el otro bloque del módulo de comunicación, procesar dichas peticiones para que sean ejecutadas por el resto de módulos del sistema, y responder al usuario con la respuesta obtenida de los servidores. Este módulo está formado por:

- Un módulo principal, ServidorPC. Se encarga de recibir todas las peticiones de las aplicaciones de los usuarios, de procesarlas y responderlas de forma satisfactoria, si es posible.

- Un sitio web, Administración. Tiene como finalidad gestionar todos los recursos con los que interacciona el módulo de comunicación, como son: la gestión de usuarios del sistema, la gestión de módulo pertenecientes al sistema, la gestión de administradores y la configuración del ServidorPC.

- Una aplicación que instancia la base de datos, Crear base de datos. La aplicación tiene como objetivo la creación de la base de datos del Servidor.
- La base de datos, Servidor. La base de datos contiene toda la información necesaria para el correcto funcionamiento de la aplicación.

Una vez que hemos diseñado la arquitectura que vamos a implementar, procederemos a describir cual es el intercambio de información entre los diferentes bloques de los que consta el módulo de comunicación desarrollado.

Primero debemos empezar por el bloque distribuido en el dispositivo móvil, destacando el intercambio de información existente entre la aplicación de usuario y el bloque implementado del módulo de comunicación.

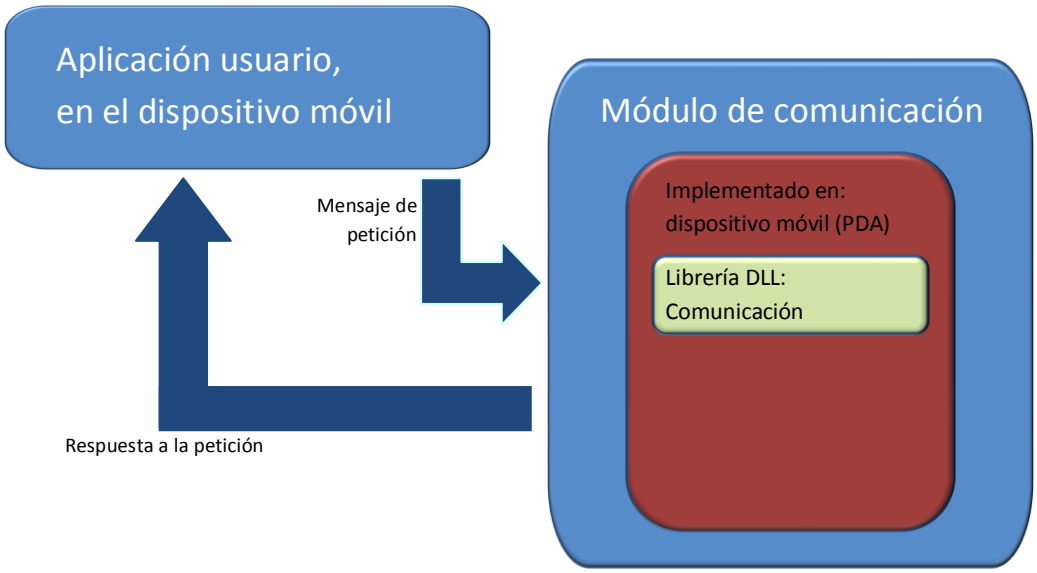


Figura 14: Intercambio de información - Dispositivo móvil y módulo de comunicación.

El formato del mensaje de petición que debe ser transmitido al módulo de comunicación tiene que tener la siguiente sintaxis, en caso contrario se producirá un error durante la comprobación del formato en la creación de la conexión con el servidor del módulo de comunicación, ServidorPC.

Sintaxis del mensaje de petición:

<direccion_ip> </direccion_ip>
<puerto> </puerto>
<tipo_comunicacion> </tipo_comunicacion>
<cod_peticion> </cod_peticion>
<usuario>
    <id> </id>
    <password> </password>
    <id_pda> </id_pda>
</usuario>
<contenido> </contenido>

El formato de la respuesta de la petición está formado por las etiquetas código de error y contenido.

- Etiqueta código de error: sus posibles valores son TRUE, en el caso de que se haya producido un error en la ejecución de la petición dentro del módulo de comunicación, o FALSE si la petición de la aplicación del usuario se ha procesado satisfactoriamente

- Etiqueta contenido: en el caso de que el valor de la etiqueta código de error sea TRUE, el valor de la etiqueta contenido será el tipo de error que se ha producido, en caso contrario, si el valor de la etiqueta código de error es FALSE, el valor de la etiqueta contenido será la respuesta a la petición realizada.

Sintaxis del mensaje de respuesta a la petición:

<cod_error> </cod_error>
<contenido> </contenido>

El siguiente intercambio de información se produce entre el bloque implementado en el dispositivo móvil y el bloque que está desplegado en el servidor del sistema.

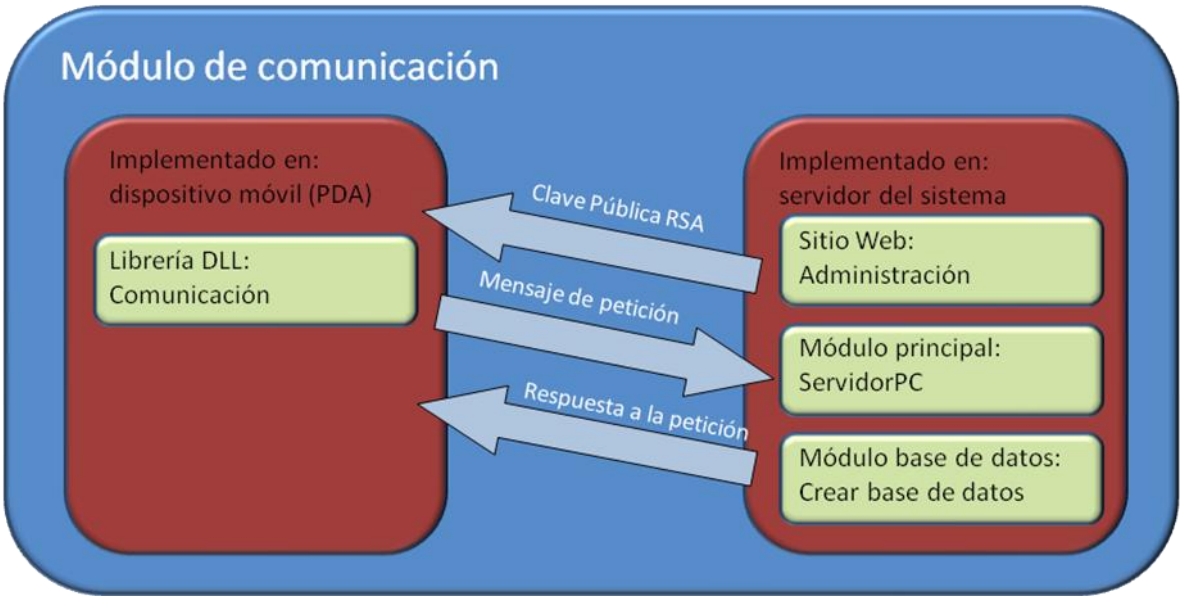


Figura 15: Intercambio de información entre el módulo de comunicación.

Tanto el formato del mensaje de petición como el formato de la repuesta a la petición, siguen siendo los mismos que los transmitidos entre la aplicación del usuario y la librería DLL, Comunicación.

En el bloque implementado en el servidor del sistema, el intercambio de información se produce con el resto de módulos pertenecientes al sistema y con la base de datos. Aunque, tenemos que tener en cuenta que el único módulo que intercambia datos con el resto de módulos es módulo principal, ServidorPC. En cambio, en el intercambio de información con la base de datos actúan todos los sub-módulos (administración, ServidorPC y crear base de datos) del módulo de comunicación implementado en el servidor.

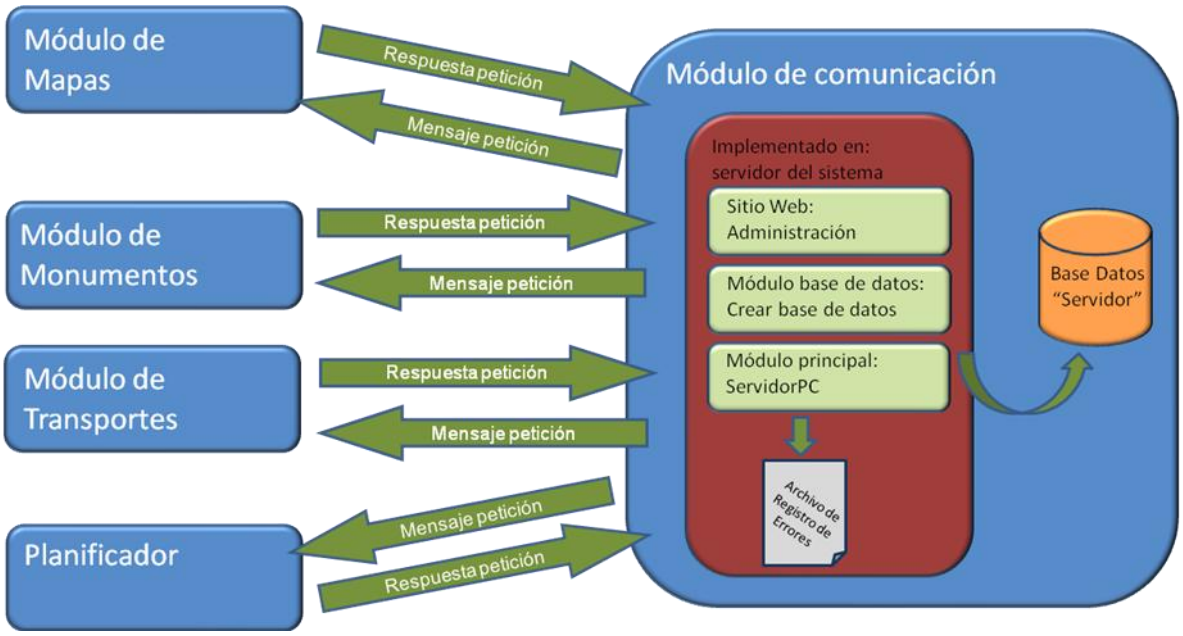


Figura 16: Intercambio de información entre el módulo de comunicación y el sistema.

Con la descripción del intercambio de información entre el módulo de comunicación y el resto de módulos del sistema terminamos la explicación sobre la arquitectura del proyecto.

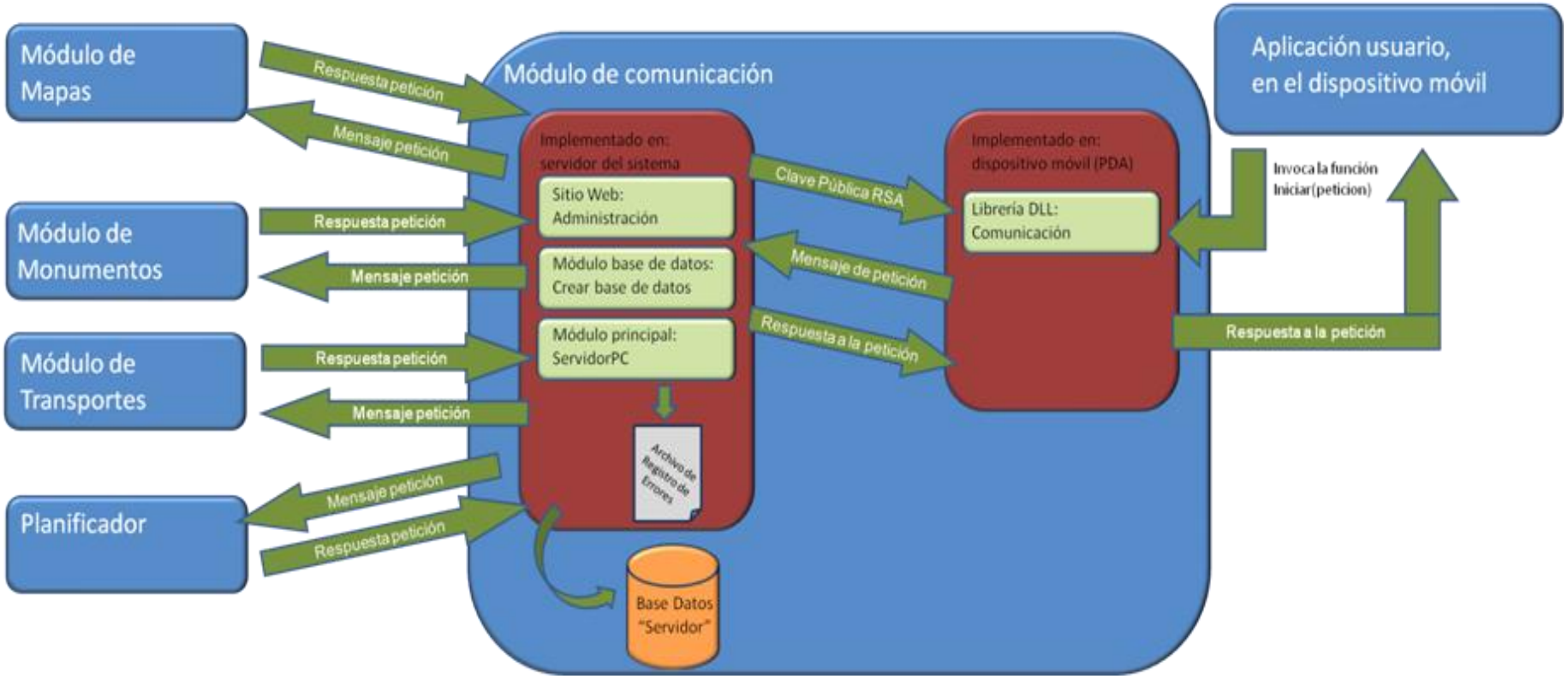


Figura 17: Intercambio de información del módulo de comunicación.

4.3. Desarrollo del proyecto

En este apartado se van a describir las técnicas y algoritmos empleados en el desarrollo e implementación del módulo de comunicación del sistema de gestión de información turística. Además, se razonarán las decisiones que se han tomado a lo largo de su desarrollo, tanto en el diseño como en la implementación realizada.

Nos ayudaremos de los distintos gráficos y diagramas empleados en el desarrollo e implementación de la aplicación.

Empezaremos describiendo el diseño del módulo de comunicación implementado en el dispositivo móvil, ya que es más simple y más fácil de comprender que el módulo desplegado en el servidor del sistema, que es bastante más complejo, debido a su estructura formada por otros tres sub-módulos o aplicaciones.

4.3.1. Diseño del módulo de comunicación implementado en el dispositivo móvil

Antes de empezar con el diseño del módulo, vamos a realizar una breve descripción del mismo, donde comentaremos sus objetivos y los sub-módulos por los que está formado.

En primer lugar, deberíamos decir que el módulo debe ser capaz de transmitir las peticiones realizadas por la aplicación del usuario al servidor del sistema, es decir, sirve de enlace entre la aplicación del usuario en la PDA y el resto de módulos del sistema.

Además, la comunicación entre ambos bloques del módulo de comunicación se produce bajo una conexión segura, con la que certificamos la confidencialidad (de forma relativa, ya que nunca se puede afirmar al cien por cien) de los datos transmitidos por la red.

Por último, tendríamos que destacar que el módulo de comunicación desplegado en el dispositivo móvil está solamente formado una librería DLL, la cual tendrá que ser utiliza por la aplicación del usuario para poder comunicarse con los servidores que ejecutarán sus peticiones.

Después de realizar una breve descripción del módulo, explicaremos el diseño y la implementación llevada a cabo en su desarrollo.

Dado que el paradigma de programación empleado en el desarrollo del proyecto es el de orientación a objetos, el diseño de los módulos se ha realizado mediante el uso de diagramas de clases. Este nos mostrará la estructura de clases existente en el módulo, cómo son las relaciones entre clase y las estructuras de herencia, e incluirá también los atributos y métodos implementados en cada una de las clases creadas.

El diagrama de clases perteneciente al módulo de comunicación desplegado en el dispositivo móvil está formado por siete clases, las cuales están relacionadas entre sí, existiendo incluso una estructura de herencia.

La estructura de herencia presente en el diagrama se debe a la abstracción que se puede realizar de los distintos tipos de mensajes que se pueden tratar, definiendo el mensaje más simple como la clase base, Mensaje, de la que heredan el resto de mensajes, que en nuestro caso son: MensajePeticion y MensajeServer.

Las siete clases existentes en el diagrama de clase son las siguientes:

- Clase Sesión: es la más importante del módulo, ya que es la que crea la conexión con el servidor y procesa el mensaje de petición de la aplicación del usuario, apoyándose en el resto de clases que componen la librería.

- Clase CifradorRSA: contiene todos los atributos y métodos necesarios para la gestión del cifrado del mensaje.

- Clase MensajeCifrado: es la representación del mensaje cifrado, que contiene los datos del mensaje cifrado (cifrado con AES), el vector de inicialización y la clave (cifrado con el sistema de cifrado RSA) del sistema de cifrado simetrico AES.

- Clase Mensaje: simboliza la abstracción más simple del mensaje. Se encarga de comprobar el formato del mensaje.

- Clase MensajePeticion: es la abstracción del mensaje de una petición. Tiene como objetivo comprobar las etiquetas del mensaje de petición.

- Clase MensajeServer: es la representación del mensaje de petición que recibe el servidor del módulo de comunicación. Su finalidad es comprobar el formato del mensaje.

- Clase Usuario: contiene la información que se refiere al usuario que ha realizado la petición.

DIAGRAMA DE CLASES,
MODULO DE COMUNICACIONES EN PDA

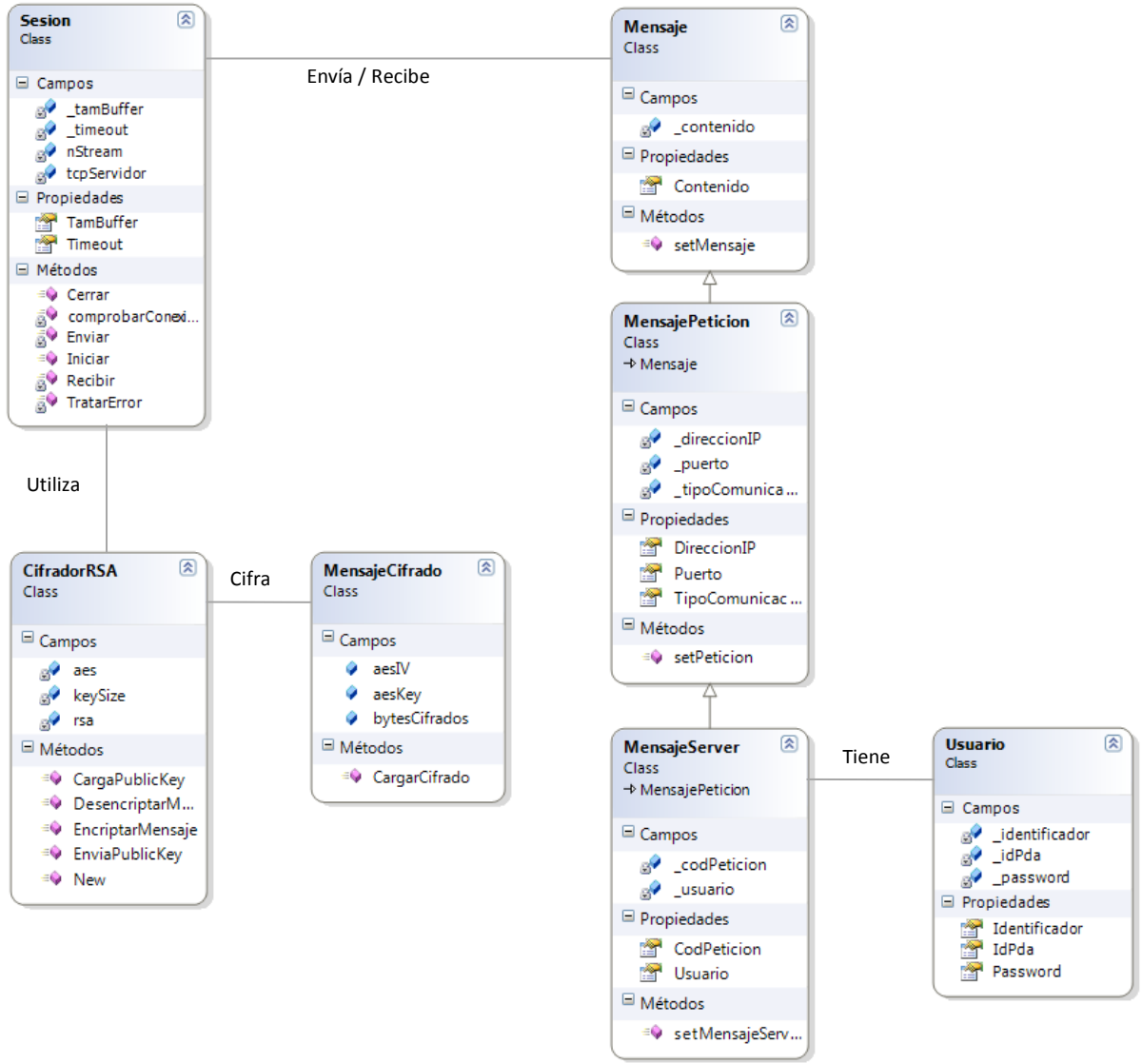


Figura 18: Diagrama de clases - La librería Comunicación.

Después de realizar una breve descripción del diagrama de clases diseñado, procedemos a describir la implementación de las principales clases del diagrama.

4.3.1.1. Descripción de las clases Mensaje, MensajePeticion y MensajeServer

La estructura de herencia que nos encontramos en la abstracción del mensaje la podríamos definir de la siguiente manera:

- La existencia de una clase base llamada Mensaje, que es la abstracción más simple, ya que solo contendría el atributo `contenido` y el método de `setMensaje`.

  Este atributo `contenido` tiene la información relativa a la petición realizada por la aplicación del usuario, que no es importante para el módulo de comunicación.

  El método `setMensaje` carga el mensaje en la clase respectiva, y además comprueba que la sintaxis de la etiqueta contenido del mensaje es correcta.

- La siguiente clase que aparece es la clase MensajePeticion que hereda de forma directa de la clase Mensaje. Esta clase MensajePeticion es la abstracción del mensaje de una petición, por este motivo posee los atributos `direccionIP`, `puerto` y `tipo de comunicación`, y el método `setPeticion`.

  El significado de los atributos es el siguiente:

  - `_direccionIP`: indica el valor de la dirección IP del servidor (del módulo de comunicación) al que hay que enviar el mensaje.
  - `_puerto`: indica el valor del puerto del servidor (del módulo de comunicación) al que hay que enviar el mensaje.
  - `_tipoComunicacion`: nos indica el tipo de comunicación (GPRS, Bluetooth,…) que quiere emplear el usuario a la hora de crear la conexión.

  El método `setPeticion` carga el mensaje en la clase respectiva, y además comprueba que la sintaxis de las etiquetas (tipo de comunicación, dirección IP y puerto) del mensaje es correcta.

- La última clase de la estructura de herencia es MensajeServer, que hereda de la clase MensajePeticion y ésta a su vez de la clase base, Mensaje.

  Esta clase MensajeServer es la abstracción del mensaje de petición que recibe el servidor del módulo de comunicación. Posee los atributos y métodos heredados de la superclase a la que se le agregan nuevos atributos y métodos.

  Los atributos propios definidos en esta clase son:

  - `_codPeticion`: identifica el tipo de petición, que utilizará el módulo de comunicaciones del Servidor Pc, para saber a qué otro módulo de la aplicación global debe enviarse dicha petición.
  - `_usuario`: este atributo es a su vez una clase, contiene la información relativa al usuario que ha realizado la petición. Los atributos que posee esta clase son:

    - `_identificador`: identifica de manera única a cada uno de los usuarios del sistema.
    - `_password`: contraseña empleada por el usuario para acceder a los servicios proporcionados por el sistema.
    - `_idPDA`: identifica de manera única el dispositivo móvil del usuario.

  El método `setMensajeServer` carga el mensaje en la clase respectiva, y además comprueba que la sintaxis de las etiquetas (usuario y código petición) del mensaje es correcta.

4.3.1.2. Descripción de las clases CifradorRSA y MensajeCifrado

Ambas clases están relacionadas con el cifrado del mensaje de petición enviado desde la aplicación del usuario. Empezaremos explicando la clase MensajeCifrado antes que la clase CifradorRSA, para que esta última sea comprendida con más facilidad.

La clase que contiene los datos del mensaje cifrado es la clase MensajeCifrado, aunque no sólo contiene los datos cifrados, sino que también tiene el vector de inicialización y la clave (se encuentra cifrada con el sistema de cifrado RSA) del sistema de cifrado simétrico AES.

La clase MensajeCifrado contiene todos los atributos y métodos necesarios para la gestión del cifrado del mensaje.

Esta clase tiene definidos los siguientes atributos:

- `bytesCifrados`: contiene los datos del mensaje cifrados mediante el sistema de cifrado simétrico AES.

- `aesKey`: contiene el valor de la clave del cifrador AES utilizado en el cifrado del mensaje. Esta clave se encuentra cifrada mediante la utilización del sistema de cifrado asimétrico de clave pública RSA.

- `aesIV`: contiene el vector de inicialización del cifrador AES utilizado en el cifrado de los datos del mensaje.

Sólo hay un método en la clase MensajeCifrado, es el siguiente:

- `CargarCifrado()`: método que carga los datos cifrados recibidos del módulo de comunicación del dispositivo móvil en la clase, para que la clase CifradorRSA tenga una implementación lo más simple posible. El método recibe como parámetros de entrada los datos que debemos cargar en esta clase. Estos datos son los recibidos del módulo del dispositivo móvil.

  Tendríamos que destacar que este método no se utiliza en el módulo de comunicación del dispositivo móvil, es decir, la librería que utiliza la aplicación del usuario no invoca en ningún momento este método.

  Sin embargo, este método si se usa en el servidor del módulo de comunicación a la hora de cargar los datos recibidos por parte del usuario, ya que estos datos sí se encuentran cifrados.

La clase que contiene toda la información necesaria para el cifrado de la información es la clase CifradorRSA. Esta clase es la segunda más importante de la librería de comunicación, por detrás de la clase Sesión, ya que es la encargada del cifrado del mensaje.

Para el cifrado de los datos la clase CifradorRSA contiene una serie de atributos y métodos propios para llevar a cabo con éxito el ocultamiento de los datos del mensaje.

Tendremos que realizar la importación de la librería de criptografía de .Net, para poder utilizar los sistemas de cifrado deseados. El nombre de la librería es: System.Security.Cryptography.

La clase CifradorRSA tiene definidos los siguientes atributos:

- `aes`: contiene toda la información relativa al sistema de cifrado simétrico AES.

- `rsa`: contiene toda la información relativa al sistema de cifrado asimétrico de clave pública RSA

- `keySize`: indica la longitud que poseerá la clave de cifrado del sistema de cifrado simétrico AES. Por defecto, su valor es de 128.

Los métodos y funciones implementadas en la clase CifradorRSA son:

- `New()`: método constructor de la clase CifradorRSA, en la que se lleva a cabo la instanciación de un nuevo objeto de la clase.

  La instanciación de este objeto consiste en:

  - La asignación de un nuevo objeto del sistema de cifrado RSA al atributo `rsa` de la clase.
  - La asignación de un nuevo objeto del sistema de cifrado AES al atributo `aes` de la clase.
  - La asignación del tamaño de la clave del cifrador AES mediante el uso del atributo `keySize` de la clase.

- `EnviarPublicKey()`: función se encarga de obtener todos los parámetros del cifrador RSA, exceptuando la clave privada.

  La función devuelve los parámetros del cifrador RSA, exceptuando la clave privada.

- `CargarPublicKey()`: método que establece las propiedades del cifrador RSA, que se pasan como parámetro de entrada al método.

- `EncriptarMensaje()`: esta función es la encargada de cifrar el mensaje de la petición mediante el uso del cifrador AES y de cifrar la clave del AES mediante el uso del cifrador RSA.

  Esta función tiene como parámetro de entrada los datos que tiene que cifrar. La función devuelve un objeto de la clase MensajeCifrado, en el que se encuentra el vector de inicialización, la clave cifrada del cifrador AES y el mensaje cifrado.

  Debido a la importancia de la función, a continuación se muestra el pseudocódigo comentado de su implementación.

Pseudocódigo:

```
función EncriptarMensaje(datos) devuelve mensajeCifrado
{
‘ generamos la clave y el vector de inicialización del cifrador AES
aes.GenerarClave()
aes.GenerarVector()
‘ ciframos la clave del cifrado AES con el cifrador RSA
claveAESCifrada = rsa.Cifrar(aes.clave)
‘ ciframos los datos del mensaje con el cifrador AES
datosCifrados = aes.Cifrar(datos)
‘ cargar los datosCifrados, claveAESCifrada y vector en MensajeCifrado
mensajeCifrado.vector = aes.vector
mensajeCifrado.clave = claveAESCifrada
mensajeCifrado.contenido = datosCifrados
return mensajeCifrado
}
```

- `DesencriptarMensaje()`: esta función es la encargada de descifrar la clave del AES mediante el uso del cifrador RSA y de descifrar el mensaje de la petición mediante el uso del cifrador AES.

  La función recibe como parámetros de entrada el objeto MensajeCifrado, en el que se encuentra la clave (cifrada con el cifrador RSA), el vector de inicialización del cifrador AES y los datos (cifrados con el cifrador AES). Esta función devuelve una cadena con los datos del mensaje sin cifrar.

  Debido a la importancia de la función, a continuación se muestra el pseudocódigo comentado de su implementación.

  Pseudocódigo:

```
función DesencriptarMensaje(mensajeCifrado) devuelve contenido
{
‘ descifframos la clave del AES
claveAES = rsa.Descifrar(mensajeCifrado.clave)
‘ establecemos los parámetros del cifrador AES
aes.clave = claveAES
aes.vector = mensajeCifrado.vector
‘ descifframos los datos con el cifrador AES
contenido = aes.Descifrar(mensajeCifrado.contenido)
return contenido
}
```

4.3.1.3. Descripción de la clase Sesión

La clase Sesión es la de mayor importancia, ya que es la encargada del establecimiento, mantenimiento y cierre de la conexión con el servidor del sistema, es decir, crea la sesión con el servidor para transmitir la petición.

Para poder realizar operaciones sobre una conexión, es necesario importar las librerías de conexión a red de .Net. El nombre de las librerías es: System.Net y System.Net.Sockets.

Los atributos de los que consta son:

- `tcpServidor`: contiene información relativa a la conexión establecida con el servidor a través de una red TCP.

- `_tamBuffer`: indica el tamaño máximo de bytes, que puede enviar o recibir el socket o el buffer de la conexión, cada vez que se realiza una lectura o escritura del socket.

- `_timeout`: indica el periodo de tiempo durante el cual `tcpServidor` esperará para recibir datos tras iniciarse una operación de lectura.

- `nStream`: proporciona los métodos necesarios para el tratamiento de la conexión establecida con el servidor.

Los métodos y funciones implementados en la clase son:

- `ComprobarConexion()`: se trata de la función que comprueba si el dispositivo móvil tiene en activo una conexión a Internet. La función devuelve True si existe dicha conexión a Internet, y False en caso contrario.

  Debido a la importancia de la función, expondremos cual ha sido la implementación escogida para realizarla, y cuál ha sido el motivo de descartar las otras opciones posibles que existían. En primer lugar, explicaremos las implementaciones descartadas, y posteriormente la elegida.

  La primera implementación descartada consistía en realizar una petición web y esperar la respuesta, en el caso de que no exista respuesta nos indica que no hay conexión a internet, aunque puede darse el caso de que el sitio web al que se realizaba la petición no estuviese funcionando en ese momento, aún siendo de alta disponibilidad.

La segunda implementación descartada consistía en la utilización de un método del sistema operativo, que nos indicaba si existía una conexión activa a internet. El problema en este caso, era que este método no está disponible en el Compact Framework .Net.

Se ha elegido la siguiente opción, porque se ha pensado que era la más óptima en este caso concreto, ya que no se necesita ningún acceso a servidores externos.

Pseudocódigo:

```
función comprobarConexion() devuelve  conexionInternet
{
nombEquipo = ObtenerNombreEquipo()
direccionIPLocal = ObtenerDireccionIP(nombEquipo)
direccionIPEquipo = ObtenerDireccionIP()
Si direccionIPLocal = direcciónIPEquipo entonces
    conexionInternet = False
Sino
    conexionInternet = True
}
```

- `Recibir()`: la función tiene como misión la recepción de los datos por parte del servidor del sistema. La función devuelve los datos recibidos.

- `Enviar()`: método que envía los datos de la petición al servidor. El método recibe como parámetros de entrada los datos a enviar al servidor.

- `Iniciar()`: es la función principal de la librería, ya que realiza el procesamiento de la sesión que se lleva a cabo con el servidor. La función tiene como parámetro de entrada el mensaje de la petición, y devuelve la respuesta del servidor o un mensaje de error si se ha producido un error interno en la ejecución de la sesión. Este es el método que tendría que invocar la aplicación del usuario para enviar sus peticiones al servidor, y éste las procese.

Debido a la importancia de esta función se va a proceder a escribir un pseudocódigo comentado de su implementación.

Pseudocódigo:

```
función Iniciar(mensaje) devuelve respuesta
{
' comprobamos que existe una conexión a internet
Si comprobarConexion() = True entonces
    ' cargamos el mensaje, comprobando si su sintaxis es correcta
    Si cargarMensaje(mensaje) = True entonces
            ' creamos la conexión con el servidor
            conectar(direccionIP, puerto)
```

```
                ' recibimos la clave pública del cifrado RSA
                clavePublica = recibir()
                ' cargamos la clave pública en el cifrador
                cifradorRSA = cargarClavePublica(clavePublica)
                ' ciframos el mensaje con AES y la clave con RSA
                mensajeCifrado = Cifrar(mensaje)
                ' enviamos el mensaje cifrado al servidor
                enviar(mensajeCifrado)
                ' esperamos a recibir la respuesta del servidor
                Respuesta = recibir()
                ' cerramos la conexión con el servidor
                Cerrar()
        Sino
                respuesta = "se ha producido un error en la sintaxis."
    Sino
        respuesta = "no existe ninguna conexión a internet"
    Return respuesta
    }
```

- `Cerrar()`: método encargado del cierre de la sesión, es decir, cierra la conexión existente entre el dispositivo móvil y el servidor.
- `TratarError()`: función encargada del tratamiento de los errores si el error se ha producido de forma interna durante la ejecución la sesión.

4.3.2. Diseño del módulo de comunicación implementado en el servidor del sistema

Antes de empezar con la explicación del diseño del módulo, vamos a realizar una breve descripción del mismo, donde comentaremos cuáles son sus principales objetivos y los sub-módulos por los que está constituido.

En primer lugar, deberíamos decir que el módulo debe tener la capacidad de recibir todas las peticiones realizadas por el otro bloque del módulo de comunicación, procesar dichas peticiones para que sean ejecutadas por el resto de módulos del sistema, y responder al usuario con la respuesta obtenida de los servidores.

Además, tendremos que tener en cuenta que la comunicación entre ambos bloques del módulo de comunicación se produce bajo una conexión segura, ya que existe una transmisión de datos privados del usuario que utiliza el sistema.

El módulo de comunicación desplegado en el servidor del sistema de gestión de información turística está formado por:

- Un módulo principal, ServidorPC. Se encarga de recibir todas las peticiones de las aplicaciones de los usuarios, de procesarlas y dar respuesta a éstas de forma satisfactoria, si es posible.

- Un sitio web, Administración. Su objetivo es la gestión de todos los recursos con los que interacciona el módulo de comunicación, como son: la gestión de usuarios del sistema, la gestión de módulos pertenecientes al sistema, la gestión de administradores y la configuración del ServidorPC.

- Una aplicación que instancia la base de datos, Crear base de datos. La aplicación tiene como finalidad la creación de la base de datos del Servidor, para que el módulo de comunicación funcione correctamente.

- La base de datos, Servidor. La base de datos contiene toda la información necesaria para el correcto funcionamiento de la aplicación, es decir, contiene los usuarios y los administradores registrados en el sistema, aparte de los módulos restantes del sistema global y la configuración del módulo principal, ServidorPC.

Después de realizar una breve descripción del módulo, explicaremos el diseño y la implementación llevada a cabo en su desarrollo.

4.3.2.1. Diseño de la base de datos, Servidor

La base de datos se encuentra desplegada en el servidor de bases de datos del sistema, que en nuestro caso es Microsoft SQL Server 2005.

La base de datos, Servidor, almacenará toda la información relativa a los usuarios, administradores y módulos del sistema, incluyendo también una tabla con la configuración del módulo principal del servidor, ServidorPC.

A continuación, realizaremos la descripción de los datos recogidos en la base de datos que se ha desarrollado en el módulo de comunicación implementado en el servidor del sistema.

La base de datos, Servidor, está constituida por el siguiente conjunto de tablas:

- Administradores: esta tabla contendrá toda la información relativa a aquellas personas que puedan modificar las configuraciones del sistema y realizar la gestión de los módulos y los usuarios del sistema.

La tabla administradores está constituida por los campos:

- `Id_administrador`: identifica de forma única a cada uno de los administradores. Nombre que se utiliza para entrar en el sistema.
- `Password`: contiene la contraseña establecida por el propio administrador, con la que se le permite entrar en el sistema.
- `Nombre`: indica el nombre del administrador. No es obligatorio introducirlo.
- `Apellido_1`: contiene el primer apellido del administrador. No es obligatorio rellenarlo cuando un administrador se da de alta.
- `Apellido_2`: contiene el segundo apellido del administrador. Tampoco es obligatorio introducirlo.
- `Email`: contiene la dirección de correo electrónico que facilitó el administrador a la hora de registrarse.

- Usuarios: esta tabla contendrá toda la información relativa a aquellas personas que puedan utilizar los servicios ofrecidos por el sistema de gestión de información turística.

  La tabla usuarios está formada por los campos:

  - `Id_usuario`: identifica de forma única a cada uno de los usuarios. Nombre que se utiliza para usar los servicios ofrecidos por el sistema.
  - `Password`: contiene la contraseña establecida por el propio usuario, con la que se le permite entrar en el sistema.
  - `Nombre`: indica el nombre del usuario. No es obligatorio introducirlo.
  - `Apellido_1`: contiene el primer apellido del usuario. No es obligatorio rellenarlo cuando un usuario se da de alta.
  - `Apellido_2`: contiene el segundo apellido del usuario. Tampoco es obligatorio introducirlo.

- `Email`: contiene la dirección de correo electrónico que facilitó el usuario a la hora de registrarse.

- `Id_PDA`: identifica de manera única el dispositivo móvil que utiliza el usuario.

- Módulos: esta tabla contiene toda la información relativa a los módulos que están implementados en el sistema de gestión de información turística, para el procesamiento de las peticiones, como son: módulo de mapas, planificador, módulo de monumentos y módulo de transportes.

  La tabla módulos posee los siguientes campos:

  - `Id_modulo`: identifica de forma única a cada uno de los módulos que forma el sistema.

  - `Direccion_ip`: contiene la dirección IP en la que está funcionando el módulo.

  - `Puerto`: contiene el puerto en el que está ejecutándose el módulo.

- Configuración: esta tabla contiene toda la información relativa a la configuración del módulo principal, ServidorPC.

  La tabla configuración está formada por los campos:

  - `Nombre`: identifica de forma única a cada uno de los servidores que forman el sistema.

  - `Descripción`: contiene una breve descripción del servidor. No es obligatorio al dar de alta un servidor.

  - `Puerto`: contiene el puerto en el que está ejecutándose el módulo.

  - `MaxConexion`: indica cual es el número de conexiones máximas que se pueden establecer con el servidor de forma simultánea.

  - `Timeout`: indica el periodo de tiempo durante el cual el servidor esperará para recibir datos tras iniciarse una operación de lectura. Por defecto, el valor de este campo es de 30000 milisegundos (ms).

Después de realizar la descripción de la base de datos, Servidor, que hemos desarrollado, vamos a exponer cuál ha sido el diseño que hemos utilizado. Nos ayudaremos de diagramas y modelos, para entender más fácilmente el diseño desarrollado.

El modelo entidad/relación de la base de datos nos da una idea de cuál es la estructura que poseerá la base de datos que vamos a implementar en el gestor de bases de datos del servidor. El modelo entidad/relación es bastante simple, ya que está constituido por cuatro tablas independientes, es decir, las cuatro tablas por la que está formada la base de datos no tienen ningún tipo de relación entre sí.

A continuación, se muestra el modelo entidad/relación de la base de datos utilizada por el módulo principal, ServidorPC.

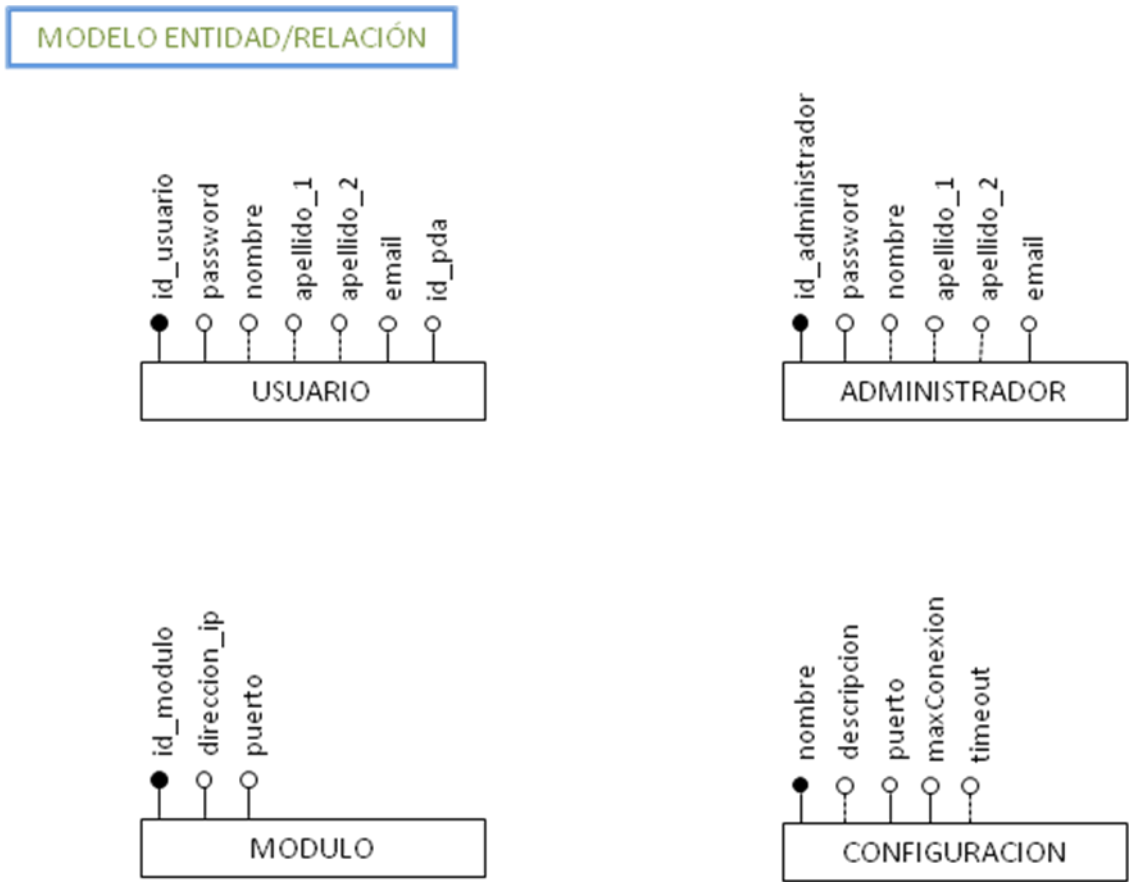


Figura 19: Modelo E/R de la base de datos Servidor.

Una vez terminado el modelo entidad/relación, pasamos a realizar el modelo conceptual de la base de datos, que contiene descripciones de las relaciones entre las entidades (en nuestro caso no existen), los atributos que poseen cada una de las entidades y las propiedades de dichos atributos.

Modelo conceptual:

Administrador (id_administrador, password, nombre*, apellido_1*, apellido_2*, email)

Usuario (id_usuario, password, nombre*, apellido_1*, apellido_2*, email, id_pda)

Modulo (id_modulo, direccion_ip, puerto)

Configuración (nombre, descripción*, puerto, maxConexion, timeout*)

Figura 20: Modelo conceptual de la base de datos Servidor.

Como con el modelo no se pueden ver reflejados los supuestos expuestos en la definición de la base de datos, a continuación se enumeran cuáles son esos supuestos no reflejados.

Solo existe un supuesto que no está reflejado en el modelo, se trata del valor por defecto que tiene asignado el campo timeout de la tabla configuración. Este valor por defecto que se le asigna al atributo timeout a la hora de insertar una configuración es de 30000 milisegundos (ms).

Después de definir el diseño de la base de datos, expondremos el script o código de debería ejecutar un gestor de base de datos (como Microsoft SQL Server 2005) para la creación de la base de datos. Aunque en nuestro caso, la creación de la base de datos la efectuará la aplicación “Crear base de datos” del módulo de comunicación del servidor del sistema.

De todos modos, explicaremos el script a utilizar, ya que existen algunos puntos de interés que tendríamos que conocer, como puede la inserción de un administrador del sistema por defecto, la inserción de la configuración del servidor por defecto e incluso el tipo de datos que poseen cada uno de los campos de las tablas.

Creación de la base de datos, Servidor, en el gestor de base de datos del sistema (Microsoft SQL Server 2005), sentencia a lanzar:

CREATE DATABASE Servidor;

Una vez creada la base de datos, procedemos a la creación de las diferentes tablas dentro de la base de datos recién creada. Las sentencias que hay que ejecutar son:

```
CREATE TABLE administrador (
        id_admin varchar(15) NOT NULL,
        password varchar(10) NOT NULL,
        nombre varchar(20),
        apellido_1 varchar(15),
        apellido_2 varchar(15),
        email varchar(30) NOT NULL,
        PRIMARY KEY (id_admin)
        );

CREATE TABLE usuario (
        id_usuario varchar(15) NOT NULL,
        password varchar(10) NOT NULL,
        id_pda varchar(15) NOT NULL,
        nombre varchar(20),
        apellido_1 varchar(15),
        apellido_2 varchar(15),
        email varchar(30) NOT NULL,
        PRIMARY KEY (id_usuario)
        );

CREATE TABLE modulo (
        id_modulo varchar(15) NOT NULL,
        direccion_ip varchar(15) NOT NULL,
        puerto varchar(5) NOT NULL,
        PRIMARY KEY (id_modulo)
        );

CREATE TABLE configuracion (
        nombre varchar(15) NOT NULL,
        descripcion varchar(200),
        puerto varchar(5) NOT NULL,
        maxConexion varchar(5) NOT NULL,
        timeout varchar(6) DEFAULT 30000,
        PRIMARY KEY (nombre)
        );
```

Tenemos que destacar que todos los campos de la tablas creadas en la base de datos, Servidor, tienen como tipo de datos “varchar()”, con esto estamos indicando que todos los atributos están definidos como cadena de caracteres.

La longitud de las cadenas viene especificada en el número indicado entre paréntesis.

Otros factores destacables serían la especificación de los campos no nulo mediante la omisión del parámetro “NOT NULL” en cada uno de los atributos que fueran necesario (por ejemplo: en la tabla configuración el atributo descripción) y la especificación del valor por defecto de 30000 en el campo timeout de la tabla configuración.

Después, de crear las tablas necesarias de la base de datos Servidor para que el módulo de comunicación del servidor del sistema funcione correctamente, hemos realizado la inserción de un administrador por defecto y la inserción de la configuración por defecto del módulo principal, ServidorPC.

El motivo de la primera inserción, el administrador por defecto, es para que el sistema no se quede bloqueado, ya que para iniciar el servidor no es necesario un administrador, pero si es necesario si se quiere modificar su configuración o realizar la gestión de los usuarios y módulos del sistema.

Sentencia ejecutada para la inserción del administrador:

```
INSERT INTO administrador VALUES (
        "adminServidor",                    ' id_administrador
        "admin",                            ' password
        "default",                          ' nombre
        "default",                          ' apellido_1
        "default",                          ' apellido_2
        "default@default.com");             ' email
```

El motivo de la inserción de la configuración por defecto es, que el servidor al iniciarse obtiene la configuración del mismo de la base de datos, y en el caso de que no exista dicha configuración el servidor daría un error y no podría iniciarse.

La sentencia para la inserción de la configuración es:

```
INSERT INTO configuracion VALUES (
        "Servidor001",                      ' nombre
        "Es el servidor principal de la aplicación.",   ' descripcion
        "8000",                             ' puerto
        "100",                              ' maxConexion
        "15000");                           ' timeout
```

Con la descripción de las sentencias empleadas para la creación e inicialización de la base de datos, Servidor, terminamos con el diseño e implementación de la base de datos desplegada dentro del módulo de comunicación del sistema de gestión de la información turística.

4.3.2.2. Diseño del módulo de creación de la base de datos

Empezaremos realizando una breve descripción del módulo, donde comentaremos sus objetivos y los sub-módulos por los que está formado.

En primer lugar, deberíamos decir que el módulo tiene el objetivo de crear una nueva base de datos dentro de la instancia "./SQLEXPRESS" del gestor de base de datos Microsoft SQL Server 2005.

La base de datos creada tendrá el mismo diseño que el expuesto en el apartado anterior, ya que la finalidad del módulo de la creación de la base de datos no es otra que la de instanciar la base de datos que utilizará el módulo de comunicación implementado en el servidor del sistema.

Aunque, por defecto la aplicación de "Crear la base de datos" tiene como instancia Microsoft SQL Server 2005, "./SQLEXPRESS", para crear la base de datos, esta podría ser modificada, al igual que el nombre de la base de datos, pero teniendo en cuenta que habría que modificar la cadena de conexión a la base de datos tanto del módulo principal (ServidorPC) como del sitio Web (Administración),ya que ambos módulos están estrechamente relacionados con la base de datos.

El diseño de los módulos se ha realizado con diagramas de clases, ya que hemos utilizado un paradigma de programación de orientación a objetos para su desarrollo. El diagrama de clases perteneciente al módulo de la aplicación de la creación de la base de datos es muy simple, ya que solo está formado por una sola clase.

La única clase del módulo es BaseDatos, que contiene toda la información, métodos y funciones necesarias para la creación de una nueva base de datos dentro del gestor de base de datos.

A continuación, se muestra el diagrama de clases del módulo de la aplicación de "Crear base de datos".

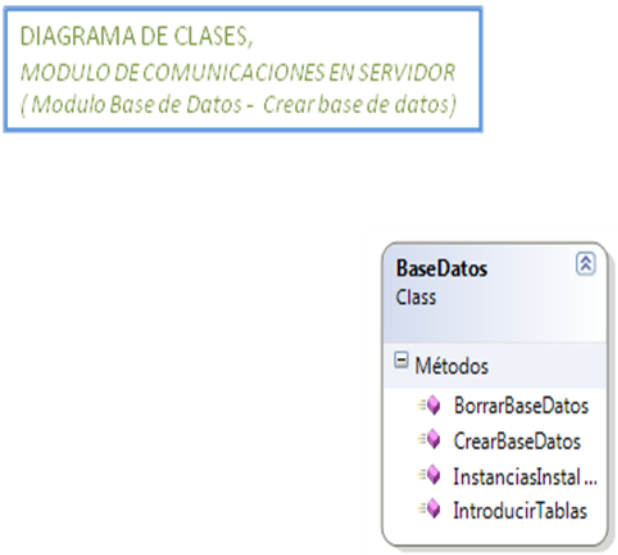


Figura 21: Diagrama de clases - Módulo "Crear base de datos".

Como ya habíamos mencionado anteriormente el diagrama de clases está formado por una única clase.

La clase BaseDatos no posee ningún atributo en su definición, pero si contiene una serie de métodos que nos ayudaran en la creación de la base de datos del servidor del módulo de comunicación. Para poder realizar las operaciones con la base de datos de forma satisfactoria, tendremos que importar la librería que nos da acceso a los gestores de base de datos de .Net. El nombre de la librería es System.Data.SqlClient.

Los métodos que contiene la clase BaseDatos son los siguientes:

- `BorrarBaseDatos()`: esta función tiene como objetivo borrar una base de datos del gestor de base de datos. La función recibe como parámetro de entrada la instancia en la que se encuentra la base de datos, y el nombre de la base de datos que deseamos que sea borrada.

  La función también posee un parámetro de salida, "mensajeError", que devuelve un mensaje de error personalizado del error que se haya producido durante la ejecución de la función, si este ha sido el caso, si no, se devolverá una cadena vacía.

  Esta función devuelve True si la base de datos, pasada como parámetro, ha sido borrada de forma satisfactoria, en caso contrario devolverá False.

- `CrearBaseDatos()`: esta función tiene como finalidad crear una base de datos dentro del gestor de base de datos. La función recibe como parámetros de entrada la instancia en la se va a crear la nueva base de datos y el nombre de la base de datos que deseamos crear.

  La función, además, tiene un parámetro de salida, "mensajeError", que devuelve un mensaje de error personalizado en el caso de que se hubiera producido algún error durante el proceso de creación de la base de datos, en el caso contrario, si todo ha ido correctamente, devolverá una cadena vacía.

  Esta función devuelve True siempre que la base de datos se haya creado correctamente, y devolverá False si se ha producido algún error durante la operación.

- `InstanciasInstaladas()`: esta función se encarga de comprobar que el gestor de bases de datos, Microsoft SQL Server, se encuentre instalado en el equipo, y de cargar las instancias del gestor siempre que éste se encuentre instalado.

La función tiene como parámetro de salida un vector de las instancias existentes. En el caso de que el gestor esté instalado, se devolverá el vector con el valor de las instancias, y en el caso de que no se encuentre instalado se devolverá un vector vacío.

Esta función devuelve True, en el caso de que el gestor de base de datos, Microsoft SQL Server, este instalado en el ordenador, y devolverá False, cuando no se encuentre instalado.

- `IntroducirTablas()`: esta función tiene el objetivo de crear todas la tablas necesarias en la base de datos, aparte de la inserción del administrador y de la configuración del servidor por defecto, para que el funcionamiento del módulo de comunicación del servidor sea correcto.

  La función recibe como parámetros de entrada la instancia de la base de datos en la que se van a insertar las tablas, y el nombre de la base de datos en la que se desea crear las tablas.

  La función, además, tiene un parámetro de salida, "mensajeError", que devuelve un mensaje de error personalizado, en el caso de que se hubiera producido algún error durante el proceso de creación de tablas y de la inserción de los datos de la base de datos, en el caso contrario, si todo ha ido correctamente, devolverá una cadena vacía.

  Esta función devuelve True siempre que las operaciones realizadas en la base de datos se hayan ejecutado correctamente, y devolverá False si se ha producido algún error durante la operación.

  Tenemos que destacar que si se produce un error en una de las operaciones durante su ejecución, serán desechadas todas las operaciones realizadas hasta ese momento, dejando el estado de la base de datos igual que cuando empezaron a ejecutarse las operaciones de creación de tablas y de inserción de datos.

Una vez terminada la descripción del diagrama de clase y de la correspondiente clase, podemos explicar en qué consiste el interfaz implementado en esta aplicación.

La interfaz desarrollada es bastante simple, ya que solo cuenta con una ventana, en la que se podrá realizar la creación de la base de datos.

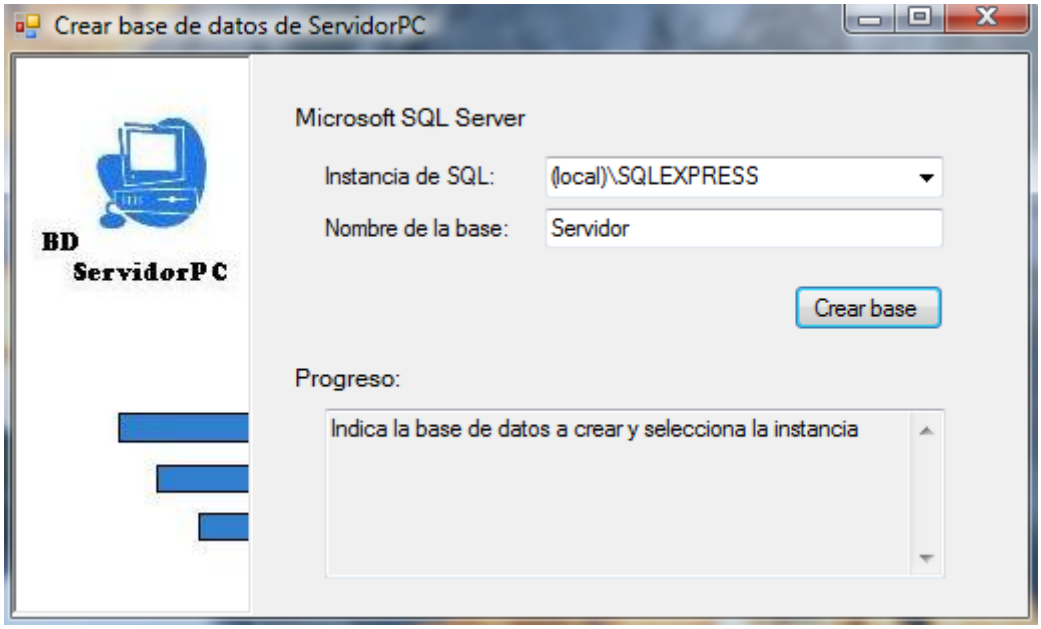


Figura 22: Interfaz de la aplicación “Crear base de datos”.

La interfaz de la aplicación es muy básica y la podemos dividir en dos pequeños bloques:

- Configuración para la creación de la base de datos, que posee el título “Microsoft SQL Server”. Está compuesto por:

    - Un desplegable, que posee el nombre de todas las instancias de SQL Server instaladas en el equipo. Por defecto, se selecciona “local\SQLEXPRESS”, ya que es la configuración por defecto que utilizará el módulo de comunicación del servidor.

      Como ya se ha comentado en el caso de modificar la instancia en la se va a crear la base de datos, se tendría que tener en cuenta la modificación de las cadenas de conexión del resto de módulos, tanto el módulo de administración como el módulo principal.

    - Una caja de texto, en la que se introducirá el nombre de la base de datos que se desea crear. Por defecto, es Servidor, ya que es la base de datos que utilizará el módulo de comunicación del servidor.

Si se modifica el nombre de la base de datos, habría que modificar el parámetro, cadena de conexión, de la configuración del módulo de comunicación del servidor.

- El botón “Crear base”, en el que al hacer clic se invoca a los métodos necesarios para la creación de la base de datos solicitada.

- Informe del progreso, que está definido con el título “Progreso”. Este apartado únicamente informa al usuario, tanto del estado en el que se encuentra la aplicación en cada momento, como de si se ha producido al algún error durante la ejecución de alguna operación.

Con la descripción de la interfaz terminamos el diseño e implementación del módulo de creación de la base de datos del servidor.

4.3.2.3. Diseño del módulo de administración

Vamos a empezar realizando una breve descripción del módulo, donde comentaremos sus objetivos y los sub-módulos por los que está formado.

Lo primero que tendríamos que destacar del módulo de administración, es que se trata de un sitio Web con el que vamos a poder realizar la gestión de los usuarios, administradores y módulos del sistema, además de modificar la configuración del servidor cada vez que sea necesario.

Los requisitos que debe cumplir el sitio Web de administración para la realización de las operaciones expuestas en el párrafo anterior son:

- Gestión de los usuarios. Deberá ser posible dar de alta nuevos usuarios en el sistema, eliminar a antiguos usuarios que ya no utilizan los servicios y poder realizar consulta de los usuarios que emplean el sistema.

- Gestión de los módulos. Deber ser capaz de dar de alta nuevos módulos que se incorporen al sistema, de eliminar antiguos módulos que ya no se utilizan en el sistema, y de consultar las propiedades de cada uno de los módulos del sistema.

- Gestión de los administradores. Es necesario que se puedan dar de alta nuevos administradores del módulo de comunicación, además, de poder dar baja a los administradores que ya no están implicados en el sistema, y de ser capaz de realizar consultas de los administradores actuales del sistema.

- Configuración del servidor. Deberá existir la posibilidad de cambiar la configuración del servidor siempre que sea necesario.

Además, otro requisito sería que el uso del módulo de administración estuviera restringido, ya que sólo aquellas personas que posean privilegios de administrador del servidor del módulo de comunicación, podrán realizar las operaciones de gestión y de configuración del servidor.

Por último, tendríamos que destacar que el módulo de administración está formado por un conjunto de páginas web dinámicas, un archivo de configuración y un conjunto de clases. Este conjunto de clases tendrá como finalidad la implementación todas las funcionalidades de las que consta el módulo de administración.

En primer lugar empezaremos con la descripción del diseño e implementación del conjunto de clases, a continuación seguiremos con el fichero de configuración ("Web.config"), terminando con el conjunto de páginas Web.

Para el diseño de este módulo también se ha utilizado el diagrama de clases como en los módulos anteriores. El diagrama de clases perteneciente al conjunto de clases del módulo de administración está formado por cuatro clases, destacando que no existe ninguna relación entre ellas.

Las cuatros clases existente en el diagrama de clase son las siguientes:

- Clase administrador: realiza las funciones relacionadas con la gestión de los administradores.

- Clase usuario: realiza las funciones relacionadas con la gestión de los usuarios.

- Clase módulo: realiza las funciones relacionadas con la gestión de los módulos.

- Clase configuración: realiza las funciones relacionadas con la actualización de la configuración del servidor.

A continuación, se muestra el diagrama de clases del conjunto de clases del módulo de administración.

DIAGRAMA DE CLASES,
MODULO DE COMUNICACIONES EN SERVIDOR
(Modulo Web - Administración)

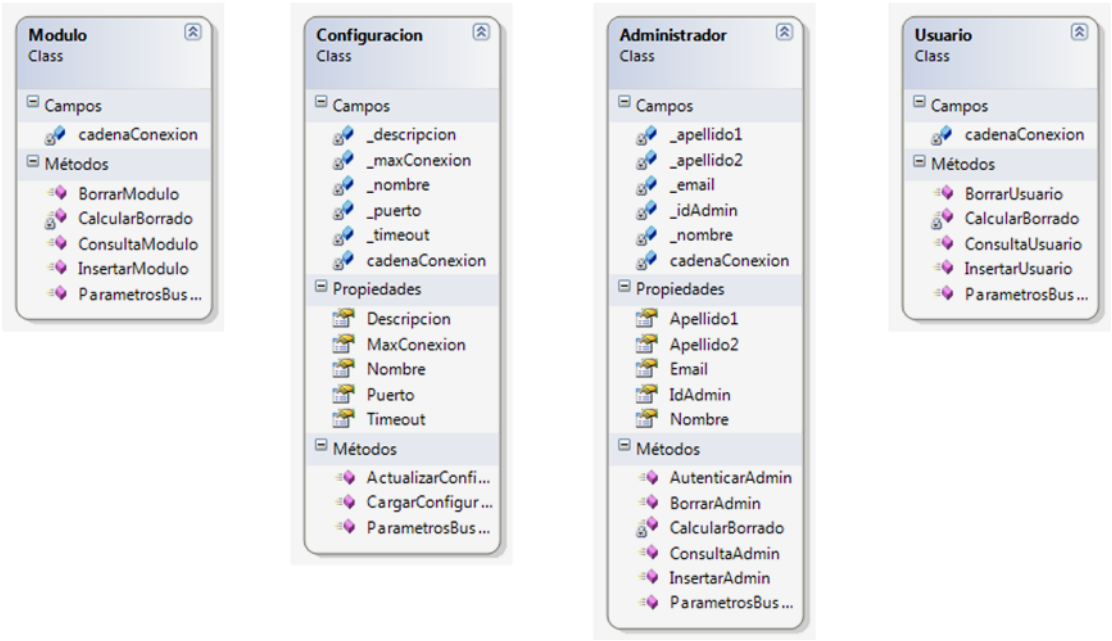


Figura 23: Diagrama de clases - Módulo de Administración.

Después de realizar una breve descripción del diagrama de clases diseñado, procedemos a describir la implementación de las principales clases del diagrama.

La estructura de las clases módulo y usuario es muy parecida, e incluso la estructura de la clase administrador también. Por eso, explicaremos las tres clases de forma simultánea.

Las tres clases constan del atributo cadenaConexion, que obtiene del fichero de configuración la cadena de conexión que utilizaremos para conectarnos a la base de datos del servidor.

Para poder realizar las operaciones con la base de datos de forma satisfactoria, tendremos que importar la librería que nos da acceso a los gestores de base de datos .Net. El nombre de la librería es System.Data.SqlClient.

Estos son los métodos o funciones de las clases:

- `CalcularBorrado()`. Función que nos indica el número de elementos que tenemos que eliminar de las tablas correspondientes, según lo que nos ha solicitado el administrador mediante la petición web.

  La función recibe como parámetro de entrada la petición web con el contenido de los datos del formulario de baja.

Esta función devuelve el número de elementos a borrar de la base de datos del servidor.

- `BorrarModulo()`, `BorrarUsuario()` y `BorrarAdmin()`. Cada una de estas funciones realiza el borrado de un elemento o varios elementos de la tabla correspondiente. En cada caso sería, `BorrarModulo()` de de la tabla modulo, `BorrarUsuario()` de la tabla usuario y `BorrarAdmin()` de la tabla administrador.

  Las funciones reciben como parámetro de entrada la petición Web realizada, que contiene todos los datos del formulario de baja.

  Estas funciones devuelven True si el borrado de los elementos se ha realizado correctamente, y False en caso contrario.

  El pseudocódigo de estas funciones es el siguiente:

```
función Borrar(peticion) devuelve estadoBorrado
{
' creamos la conexión a la base de datos
conexión = CrearConexion(cadenaConexion)
' obtenemos el número de elementos a borrar
numBorrado = CalcularBorrado()
' abrimos la conexión
conexión.Abrir()
si numBorrado > 1 entonces
        ' existe más de un elemento y entonces hay que abrir una transacción
        conTrans = conexión.AbrirTransaccion()
        ' ejecutamos la sentencias
        contador = 0
        mientras contador < numBorrado hacer
            conTrans.EjecutarSentencia("DELETE FROM xxxx WHERE xxx")
            contador = contador + 1
        ' cerramos la transacción
        conTrans.Close()
sino
        ' solo existe un elemento a eliminar, entonces ejecutamos la sentencia
        conexión.EjecutarSentencia("DELETE FROM xxxx WHERE xxx")
' cerramos la conexión
conexion.Close()
' devolvemos TRUE si no habido ningún error, sino devolveremos FALSE
return estadoBorrado
}
```

- `ParametrosBusqueda()`. Función que tiene como finalidad obtener la cadena de condiciones de la consulta solicitada por el administrador para la ejecución de la consulta.

  La función recibe como parámetro de entrada la petición web realizada por el administrador, que contendrá la información del formulario de consulta.

Esta función devuelve la cadena de condiciones de la consulta, si éstas existen, o en caso contrario, devolverá una cadena vacía.

- `ConsultaModulo()`, `ConsultaUsuario()` y `ConsultaAdmin()` Cada uno de los métodos realiza una consulta sobre la tabla correspondiente y devuelve los resultados. En cada caso sería, `ConsultaModulo()` de la tabla módulo, `ConsultaUsuario()` de la tabla usuario y `ConsultaAdmin()` de la tabla administrador.

  Las funciones reciben como parámetro de entrada la petición Web realizada, que contiene todos los datos del formulario de consulta.

  Estas funciones devuelven los datos de la consulta ejecutada contra la base de datos.

  El pseudocódigo de estas funciones es el siguiente:

```
función Consultar(peticion) devuelve datos
{
' creamos la conexión a la base de datos
conexión = CrearConexion(cadenaConexion)
' crear el adaptador de la conexión, para no tener que abrir y cerrar la conexión
adaptador = CrearAdaptador()
' obtenemos los parámetros de la consulta y creamos la sentencia
parámetros = parametrosBusqueda()
si parámetros = "" entonces
    sentencia = "SELECT FROM xxx"
sino
    sentencia = "SELECT FROM xxx WHERE" + parametros
' creamos el comando
comando = CrearComando(sentencia,conexion)
' cargamos el comando en el adaptador
adaptador.Comando = CargarComando(comando)
' creamos la estructura de datos en la que recibiremos los datos de la consulta
datos = CrearDatos()
' ejecutamos la consulta con el adaptador y cargamos los datos
 adaptador.Ejecutar(datos)
' devolvemos los datos
return datos
}
```

- `InsertarModulo()`, `InsertarUsuario()` e `InsertarAdmin()`. Cada una de estas funciones lo único que realiza es la inserción de un elemento en la tabla correspondiente. En cada caso sería, `InsertarModulo()` de de la tabla modulo, `InsertarUsuario()` de la tabla usuario e `InsertarAdmin()` de la tabla administrador.

  Las funciones reciben como parámetro de entrada la petición Web realizada, que contiene todos los datos del formulario de alta.

  Estas funciones devuelven True si la inserción del elemento se ha realizado correctamente, y False en caso contrario.

El pseudocódigo de estas funciones es el siguiente:

```
función Insertar(peticion) devuelve estadoInsetado
{
‘ creamos la conexión a la base de datos
conexión = CrearConexion(cadenaConexion)
‘ creamos la sentencia a lanzar contra la base de batos con los datos de la
peticion
Sentencia = “INSERT INTO xxx VALUES (xxx)”
 ‘ abrimos la conexión
conexión.Abrir()
 ‘ ejecutamos la sentencia
conexion.EjecutarSentencia(sentencia)
‘ cerramos la conexión
conexion.Close()
‘ devolvemos TRUE si no habido ningún error, sino devolveremos FALSE
return estadoBorrado
}
```

Concluida la descripción de las clases Modulo y Usuario, expondremos los atributos de la clase Administrador no definidos en las anteriores:

- `_idAdmin`: atributo que contiene el identificador del administrador que se encuentra con la sesión activada en el módulo de administración.
- `_nombre`: contiene el nombre del administrador que tiene abierta una sesión en el sitio Web.
- `_apellido1`: contiene el primer apellido del administrador que posee la sesión abierta.
- `_apellido2`: contiene el segundo apellido del administrador que posee la sesión abierta.
- `_email`: atributo que contiene la dirección de correo electrónico del administrador.

Estos atributos se definen en la clase, ya que se utilizan para el registro de las sesiones que se abren con el sitio Web, es decir, con el módulo de administración.

Los métodos y funciones definidas en la clase Administrador, y no comentadas anteriormente son, únicamente la función `AutenticarAdmin()`, ya que la clase Administrador es la única que necesita realizar una autenticación, a la hora del inicio de sesión de un administrador.

La función `AutenticarAdmin()` tiene como finalidad comprobar que la persona que intenta acceder al sitio Web (módulo de administración) está registrado como administrador del módulo de comunicación.

La función presenta como parámetros de entrada el id del administrador y la contraseña del administrador que hay que autenticar.

Esta función devuelve un 1 si la autenticación se ha realizado satisfactoriamente, y devolverá cualquier otro valor siempre que la identificación del administrador haya sido incorrecta.

El pseudocódigo de esta función es el siguiente:

```
función AutenticarAdmin(id,pass) devuelve numero
{
‘ creamos la conexión a la base de datos
conexión = CrearConexion(cadenaConexion)
‘ crear el adaptador de la conexión, para no tener que abrir y cerrar la conexión
adaptador = CrearAdaptador()
‘ creamos la sentencia de consulta
Sentencia = “SELECT FROM administrador WHERE id_admin =” + id + “AND password =” + pass
 ‘ creamos el comando
comando = CrearComando(sentencia,conexion)
‘ cargamos el comando en el adaptador
adaptador.Comando = CargarComando(comando)
‘ creamos la estructura de datos en la que recibiremos los datos de la consulta
datos = CrearDatos()
‘ ejecutamos la consulta con el adaptador y devuelve el numero de tuplas afectadas
numero = adaptador.Ejecutar(datos)
‘ devolvemos el numero de tuplas a las que ha afectado la consulta
return numero
}
```

La última clase que nos queda por explicar es la clase Configuración. Esta clase Configuración tiene definidos todos los atributos, métodos y funciones necesarios para poder realizar la correcta modificación de la configuración del servidor del módulo de comunicación.

Los atributos que contiene la clase Configuración son los expuestos a continuación:

- `_nombre`: contiene el nombre identificador del servidor al que vamos a acceder para cambiar su configuración.

- `_descripcion`: contiene la descripción del servidor en el que vamos a realizar las modificaciones.

- `_puerto`: contiene el puerto en el que está escuchando el servidor.

- `_maxConexion`: contiene el número de conexiones máximas que puede atender el servidor de forma simultánea.

- `_timeout`: contiene el valor del periodo de tiempo durante el cual el servidor esperará para recibir datos tras iniciarse una operación de lectura.

- `CadenaConexion`: atributo que obtiene su valor del parámetro cadena de conexión perteneciente al fichero de configuración, que utilizaremos para conectarnos a la base de datos del servidor.

La clase Configuración tiene definidos los siguientes métodos y funciones, con los que lleva a cabo la actualización de las configuración del servidor.

- `ParametrosBusqueda()`: esta función tiene como finalidad obtener la cadena de condiciones de la actualización solicitada por el administrador para la ejecución de la actualización de la configuración.

La función recibe como parámetro de entrada la petición web realizada por el administrador, que contendrá la información del formulario de actualización.

Esta función devuelve la cadena de condiciones de la actualización, si estas existen, o en caso contrario, una cadena vacía.

- `CagarConfiguracion()`: esta función realiza la consulta de la configuración del servidor solicitado por el administrador y carga los datos obtenidos en la clase, siempre que la ejecución de la consulta haya tenido éxito.

  La función recibe como parámetro de entrada el nombre del servidor que ha solicitado el administrador.

  Esta función devuelve True si existe la configuración del servidor en la base de datos, y False en caso contrario.

  El pseudocódigo de esta función es el siguiente:

```
función CargarConfiguracion(nombre) devuelve estado
{
' creamos la conexión a la base de datos
conexión = CrearConexion(cadenaConexion)
' crear el adaptador de la conexión, para no tener que abrir y cerrar la conexión
adaptador = CrearAdaptador()
'  creamos la sentencia
sentencia = "SELECT FROM configuracion WHERE nombre = " + nombre
' creamos el comando
comando = CrearComando(sentencia,conexion)
' cargamos el comando en el adaptador
adaptador.Comando = CargarComando(comando)
' creamos la estructura de datos en la que recibiremos los datos de la consulta
datos = CrearDatos()
' ejecutamos la consulta con el adaptador y cargamos los datos
numTupla = adaptador.Ejecutar(datos)
si numTupla = 1 entonces
    ' cargamos los datos de la configuración en la clase y devolvemos TRUE
    cargarDatosClase(datos)
    estado = TRUE
Sino
    ' devolvemos FALSE, porque el resultado de la consulta es incorrecto
        estado = FALSE
return estado
}
```

- `ActualizarConfiguracion()`: esta función realiza la actualización de la configuración del servidor solicitado por el administrador. La función recibe como parámetro de entrada la petición Web realizada, contiene todos los datos del formulario de actualización.

Esta función devuelve True si la actualización de la configuración del servidor se ha realizado correctamente, y False en caso contrario.

El pseudocódigo de esta función es el siguiente:

```
función ActualizarConfiguracion(peticion) devuelve estadoActualizacion
{
‘ creamos la conexión a la base de datos
conexión = CrearConexion(cadenaConexion)
 ‘ obtenemos los parámetros de la actualización y creamos la sentencia
parámetros = parametrosBusqueda()
si parámetros = “” entonces
    estadoActualizacion = TRUE
sino
    sentencia = ”UPDATE configuración SET” + parámetros + “WHERE nombre =”
            + petición.nombre
‘ abrimos la conexión
conexión.Abrir()
‘ ejecutamos la sentencia
numTupla = conexion.EjecutarSentencia(sentencia)
‘ cerramos la conexión
conexion.Close()
si numTupla = 1 entonces
    ‘ la actualización ha sido correcta, cargamos los datos y devolvemos TRUE
    CargarDatosClase(petición.datos)
    estadoActualizacion = TRUE
sino
    ‘ actualización incorrecta, ya que la operación ha afectado a varias tuplas
    estadoActualizacion = FALSE
return estadoActualizacion
}
```

Una vez terminada la explicación del diseño e implementación del conjunto de clases del módulo de administración, vamos a realizar la descripción del archivo de configuración, “Web.config”, utilizado por el módulo de administración.

El archivo de configuración, “Web.config”, sirve para definir las propiedades configurables que poseerá el sitio Web. Las propiedades que hemos establecido para cumplir algunos de los requisitos del módulo de comunicación, son:

- La creación del parámetro de la cadena de conexión, con la que nos permitirá conectarnos con la base de datos, siempre que lo necesitemos.

  Para definir la cadena de conexión en el fichero de configuración utilizaremos las siguientes etiquetas:

```
<!--
Esta etiqueta especifica las cadenas de conexión que tiene asociadas el sitio web.
 -->
<connectionStrings>
<!--
Con la siguiente etiqueta eliminamos la cadena de conexión “conexionSqlServer”.
 -->
    <remove name="conexionSqlServer"/>
```

```
<!--
Añadimos la cadena de conexión que vamos a utilizar para conectarmos a la base de dato.
 -->
    <add name="conexionSqlServer" connectionString="Data
    Source=.\SQLEXPRESS; Initial Catalog=Servidor; Integrated Security=True;"
    providerName="System.Data.SqlClient"/>
 </connectionStrings>
```

- La definición de la política de acceso al módulo de comunicación se ha llevado a cabo mediante el empleo de formularios de autenticación.

  En primer lugar, hemos denegado el acceso al sitio Web a todos los usuarios que estuviesen identificados en la sesión como anónimos.

  La etiqueta empleada para prohibir el acceso al sitio Web es:

```
<authorization>
<!--
la etiqueta deny,  con el parámetro users="?" denegamos el acceso de las paginas Web a los usuarios anonimos
 -->
        <deny users="?" />
</authorization>
```

  Una vez denegado el acceso, establecemos como método de autenticación el uso de formulario, indicándole cual es la página con el formulario de login, y la página de inicio, que se cargará si la autenticación ha sido correcta.

  La propiedad se ha especificado mediante el uso de las siguientes etiquetas:

```
<authentication mode="Forms">
<!--
Especificamos el formulario de login con el atributo "loginUrl" y la página de inicio (que se cargará cuando la autentificación sea correcta) con el atributo "defaultUrl".
-->
    <forms loginUrl="/login.aspx" defaultUrl="/index.aspx" />
</authentication>
```

Para terminar con la descripción del diseño y de la implementación del módulo de administración, vamos a explicar la estructura, las capacidades de navegación, las características de presentación, la interacción y el funcionamiento de las páginas web dinámicas del módulo.

Empezaremos identificando los componentes (nodos) y relaciones estructurales (agregación, generalización-especialización) de las que consta el módulo de administración. Esto lo vemos reflejado en el diagrama estructural, que se muestra en la siguiente página.

El diagrama nos muestra que el sitio Web está formado por dos páginas, *login* e *index*, esta última a su vez está constituida por las páginas *cerrar sesión administrador*, *cabecera*, *pie de página*, *menú* y *principal*.

Además, la página principal es un nodo compuesto, con esto queremos decir que está formado por otros nodos mediante una relación estructural de generalización-especificación. Los nodos que forman parte de la relación son: los *formularios de alta*, *de baja* y *de consulta*, *política de privacidad*, *portada*, *configuración server* y *configuración server actual*. Destacando también, la existencia de otras relaciones estructurales de generalización-especificación dentro de los nodos de los formularios.

Diagrama Estructural

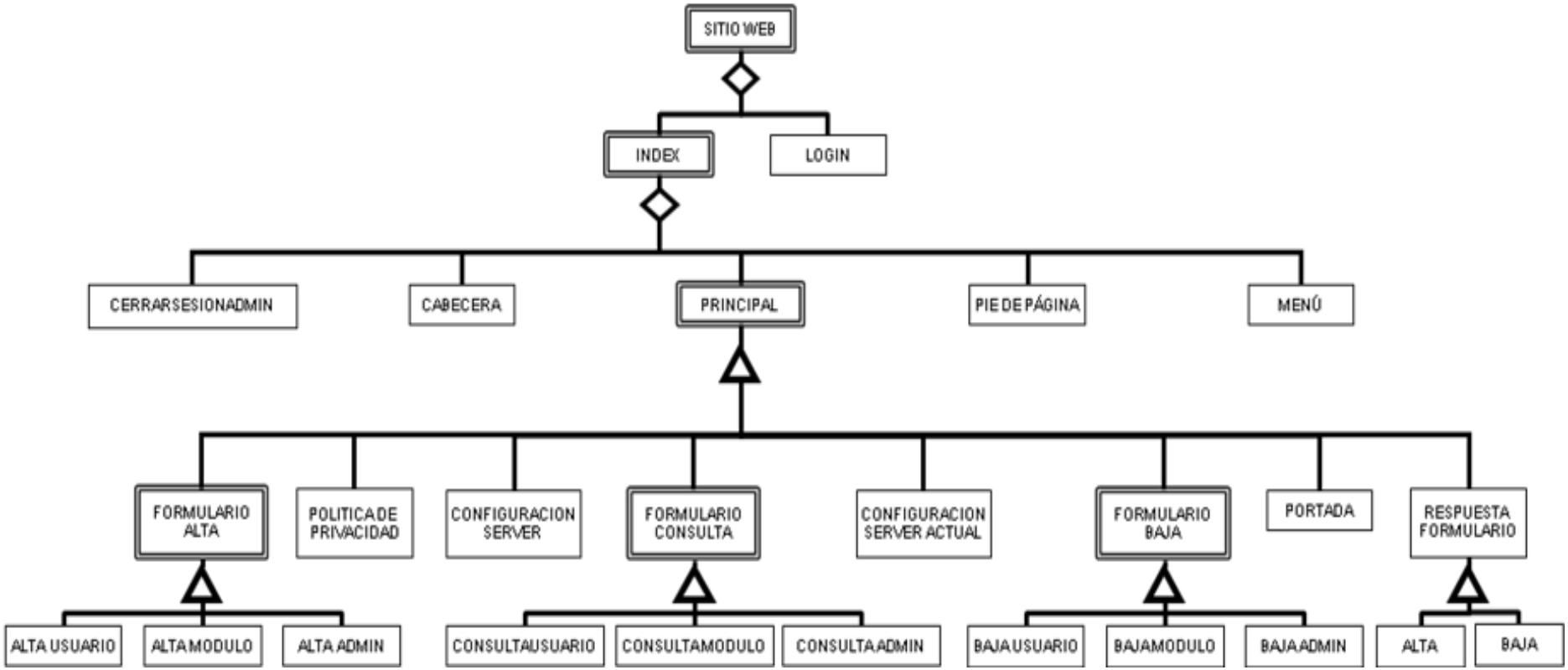


Figura 24: Diagrama Estructural - Módulo Administración.

Después de realizar el diagrama estructural del módulo de administración, procedemos a exponer cual es la capacidad de navegación dentro del sitio web.

El diagrama empleado para mostrar la estructura de navegación del módulo es el diagrama de navegación. Está dividido en dos bloques, debido a la alta densidad de figuras que existe en el diagrama.

Un primer bloque, muestra la navegación que existe entre los nodos *login* e *index* (cuando se produce el inicio de sesión de un administrador en el sitio web), y la navegación dentro del nodo de *política de privacidad*.

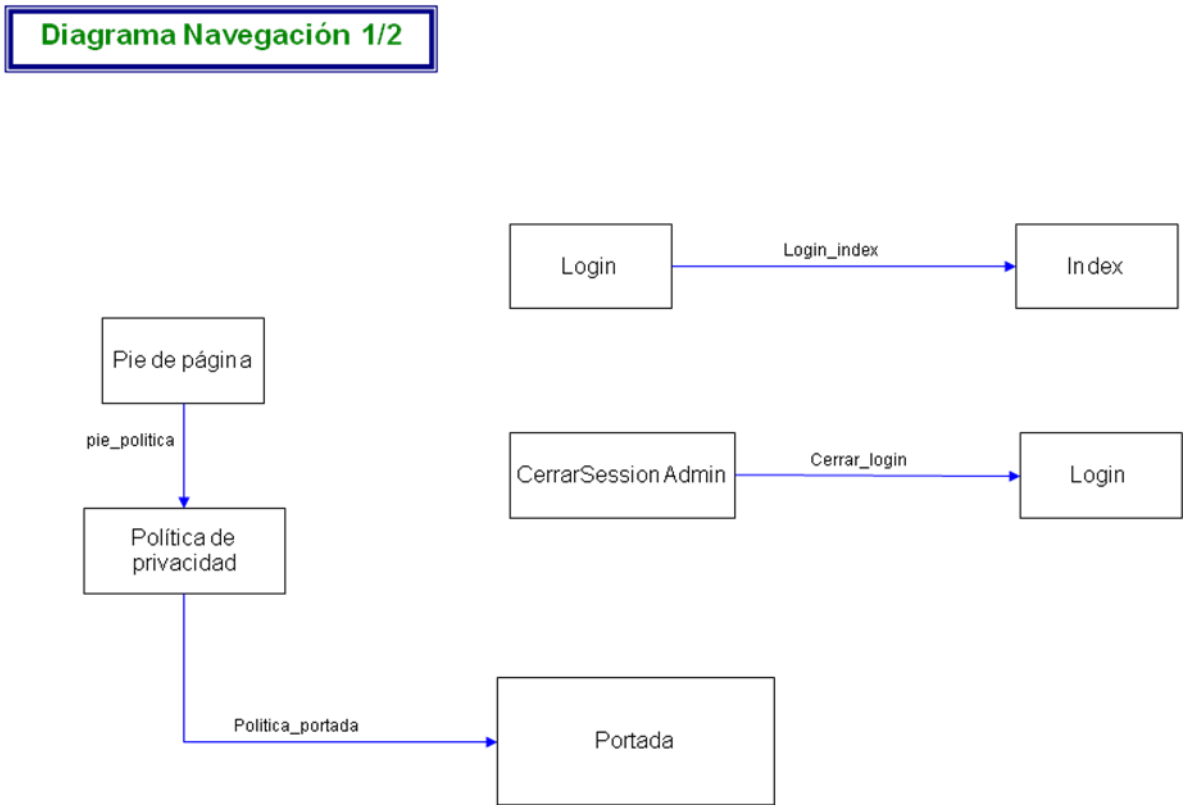


Figura 25: Diagrama de navegación 1/2 - Módulo de Administración.

En el segundo bloque del diagrama de navegación se observa como existe una herramienta de navegación, desde la que se puede ir a cualquier página de nuestro módulo de administración. Esta herramienta de navegación no es otra que el nodo *menú* del sitio Web.

También, podemos observar que desde cualquier página del sitio Web podemos volver a la página principal del sitio. Esto es debido a la barra de histórico que se encuentra en la parte superior de las páginas, que nos muestra cómo llegar desde la página *principal* a la actual y cómo volver atrás.

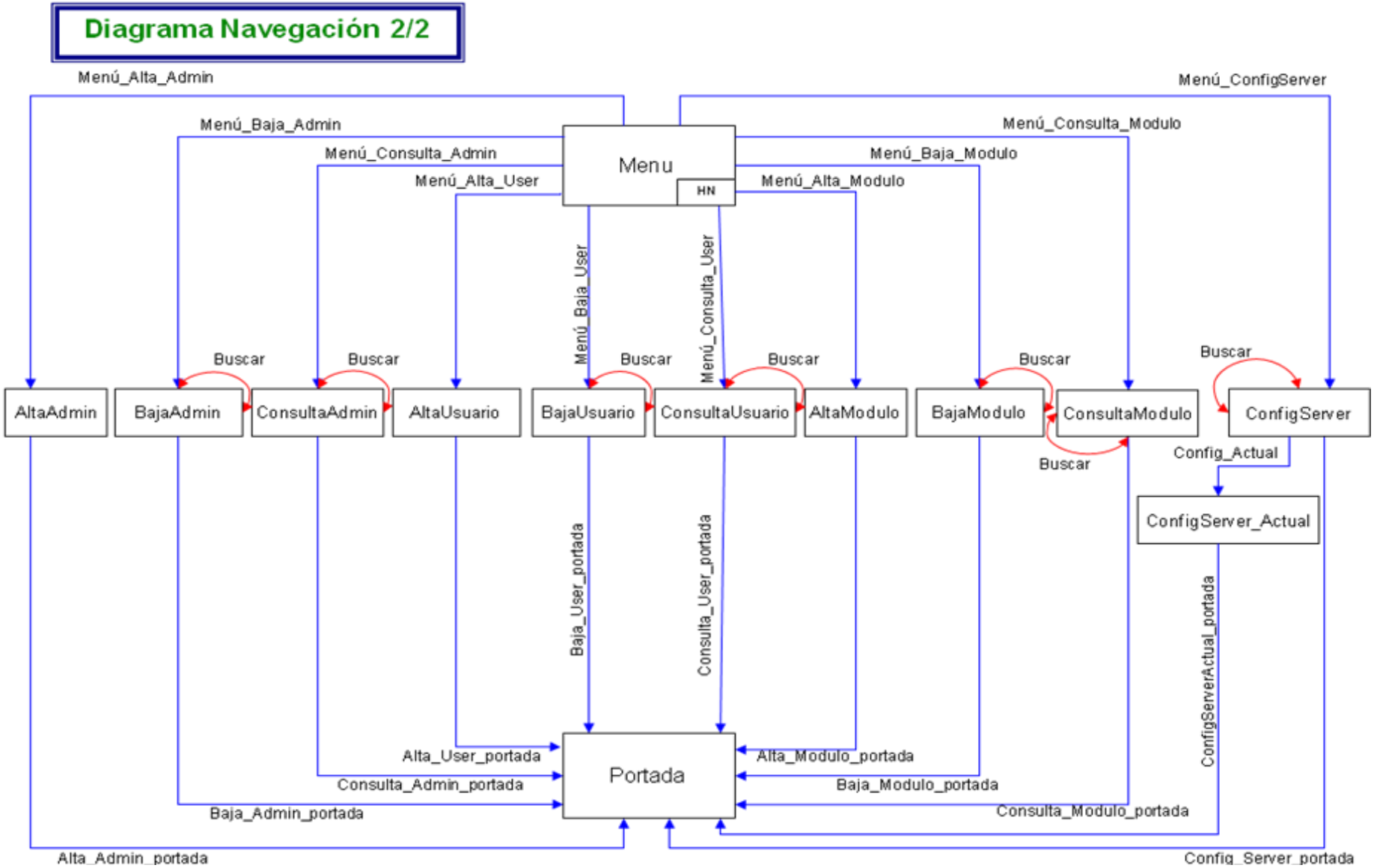


Figura 26: Diagrama de navegación 2/2 - Módulo de Administración.

Después, de describir la capacidad de navegación del sitio Web, detallaremos el funcionamiento del módulo, lo que realizaremos mediante la utilización de tablas en las que recogeremos las especificaciones funcionales.

Los campos de las tablas que recogerán la información de la especificación son:

- ID: establece un valor de identificación único para cada una de las funciones.
- NAME: indica el nombre que le hemos dado a la función correspondiente.
- TYPE: especificamos el tipo de operación que va a realizar la función definida.
- DESCRIPTION: campo en el que se realiza una breve descripción de la operación que se realizará durante la ejecución de la función.
- RELATED FUNCTION: muestra qué funciones están relacionadas con la función definida.
- RELATED EVENTS: muestra los eventos con los que está relacionada la función especificada.

Especificación de Funciones

ID:	F001
NAME:	Validación de formulario de Alta Admin
TYPE:	Verificación
DESCRIPTION:	Comprueba los datos introducidos en los campos del formulario del administrador.
RELATED FUNCTIONS:	
RELATED EVENTS:	E003

ID:	F002
NAME:	Borrar el formulario de Alta Admin
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del administrador.
RELATED FUNCTIONS:	
RELATED EVENTS:	E004

ID:	F003
NAME:	Añadir administrador
TYPE:	Actualización BBDD
DESCRIPTION:	Añade a la bbdd la información sobre un nuevo administrador
RELATED FUNCTIONS:	F004, F005
RELATED EVENTS:	E003

ID:	F004
NAME:	Borrar administrador
TYPE:	Actualización BBDD
DESCRIPTION:	Borra el administrador de la BBDD
RELATED FUNCTIONS:	F003, F005
RELATED EVENTS:	E007

ID:	F005
NAME:	Consultar administrador
TYPE:	Consulta BBDD
DESCRIPTION:	Consulta la información de los administrador es de la bbdd.
RELATED FUNCTIONS:	F003, F004
RELATED EVENTS:	E005, E009

ID:	F006
NAME:	Inicia sesión
TYPE:	Validación
DESCRIPTION:	Comprueba que el administrador exista y que la password sea correcta, para iniciar sesión.
RELATED FUNCTIONS:	F007
RELATED EVENTS:	E001

ID:	F007
NAME:	Cerrar sesión
TYPE:	Desconexión
DESCRIPTION:	Cierra la sesión del administrador.
RELATED FUNCTIONS:	F006
RELATED EVENTS:	E002

ID:	F008
NAME:	Validación de formulario de Alta Usuario
TYPE:	Verificación
DESCRIPTION:	Comprueba los datos introducidos en los campos del formulario del usuario.
RELATED FUNCTIONS:	
RELATED EVENTS:	E012

ID:	F009
NAME:	Borrar el formulario de Alta Usuario
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del usuario.
RELATED FUNCTIONS:	
RELATED EVENTS:	E013

ID:	F010
NAME:	Añadir usuario
TYPE:	Actualización BBDD
DESCRIPTION:	Añade a la bbdd la información sobre un nuevo usuario
RELATED FUNCTIONS:	F011, F012
RELATED EVENTS:	E012

ID:	F011
NAME:	Borrar usuario
TYPE:	Actualización BBDD
DESCRIPTION:	Borra el usuario de la BBDD
RELATED FUNCTIONS:	F010, F012
RELATED EVENTS:	E016

ID:	F012
NAME:	Consultar usuario
TYPE:	Consulta BBDD
DESCRIPTION:	Consulta la información de los usuario es de la bbdd.
RELATED FUNCTIONS:	F010, F011
RELATED EVENTS:	E014, E017

ID:	F013
NAME:	Validación de formulario de Alta Modulo
TYPE:	Verificación
DESCRIPTION:	Comprueba los datos introducidos en los campos del formulario del modulo.
RELATED FUNCTIONS:	
RELATED EVENTS:	E019

ID:	F014
NAME:	Borrar el formulario de Alta Modulo
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del modulo.
RELATED FUNCTIONS:	
RELATED EVENTS:	E020

ID:	F015
NAME:	Añadir modulo
TYPE:	Actualización BBDD
DESCRIPTION:	Añade a la bbdd la información sobre un nuevo modulo
RELATED FUNCTIONS:	F016, F017
RELATED EVENTS:	E019

ID:	F016
NAME:	Borrar modulo
TYPE:	Actualización BBDD
DESCRIPTION:	Borra el modulo de la BBDD
RELATED FUNCTIONS:	F015, F016
RELATED EVENTS:	E023

ID:	F017
NAME:	Consultar modulo
TYPE:	Consulta BBDD
DESCRIPTION:	Consulta la información de los modulo es de la bbdd.
RELATED FUNCTIONS:	F015, F016
RELATED EVENTS:	E021, E024

ID:	F018
NAME:	Consultar configuración
TYPE:	Consulta BBDD
DESCRIPTION:	Consulta la información de la configuración del servidor.
RELATED FUNCTIONS:	F019
RELATED EVENTS:	E010

ID:	F019
NAME:	Actualizar configuración
TYPE:	Actualización BBDD
DESCRIPTION:	Actualiza en la bbdd la información sobre la configuración del servidor.
RELATED FUNCTIONS:	F018
RELATED EVENTS:	E011

ID:	F020
NAME:	Borrar el formulario de Baja Admin
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del administrador.
RELATED FUNCTIONS:	
RELATED EVENTS:	E006

ID:	F021
NAME:	Borrar el formulario de Consulta Admin
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del administrador.
RELATED FUNCTIONS:	
RELATED EVENTS:	E008

ID:	F022
NAME:	Borrar el formulario de Baja Usuario
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del usuario.
RELATED FUNCTIONS:	
RELATED EVENTS:	E015

ID:	F023
NAME:	Borrar el formulario de Consulta Usuario
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del usuario.
RELATED FUNCTIONS:	
RELATED EVENTS:	E018

ID:	F024
NAME:	Borrar el formulario de Baja Modulo
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del modulo.
RELATED FUNCTIONS:	
RELATED EVENTS:	E022

ID:	F025
NAME:	Borrar el formulario de Consulta Modulo
TYPE:	Resetear
DESCRIPTION:	Borra los datos introducidos en los campos del formulario del modulo.
RELATED FUNCTIONS:	
RELATED EVENTS:	E025

Tabla 1: Especificación de funciones – Módulo de Administración.

El siguiente paso que deberíamos dar para realizar un diseño y una implementación adecuada del módulo de administración sería definir la estructura interna de todos los nodos de nuestro sitio Web, tanto simples como compuestos.

El diagrama que utilizaremos para realizar esta tarea será el diagrama interno de nodos y contenidos.

Dada la complejidad de algunos de los nodos hemos tenido que realizar no sólo el diagrama interno y el diagrama de contenidos sino también el diagrama interno de contenidos.

Diagrama interno de nodos y de contenidos

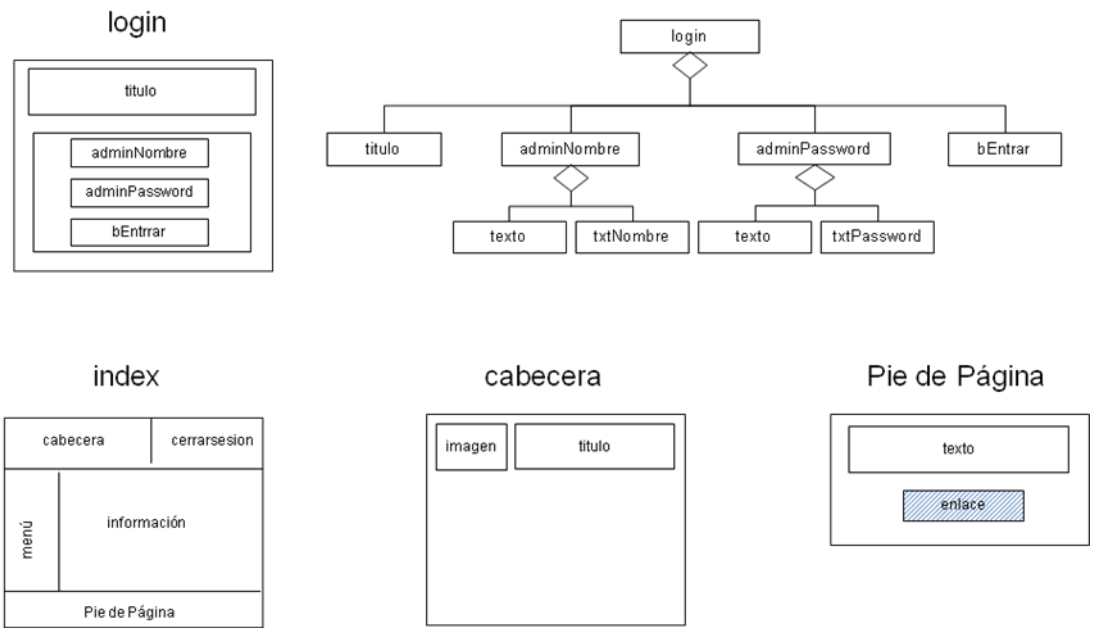


Figura 27: Diagrama interno de nodos y de contenidos 1/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

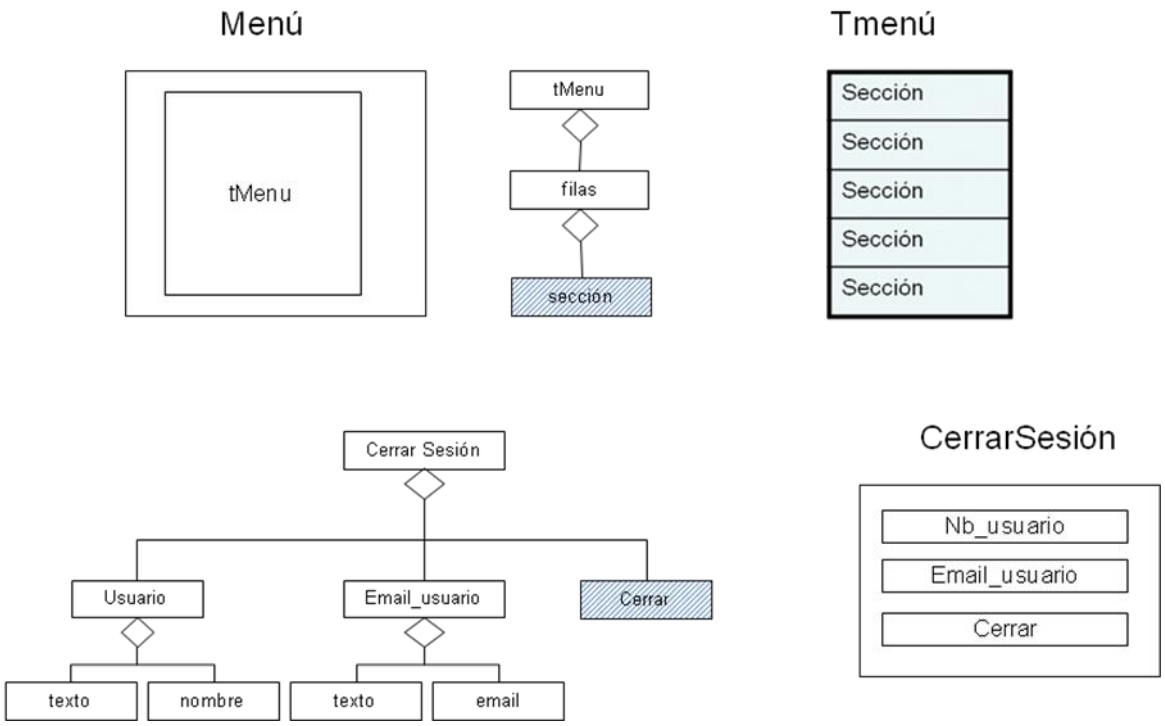


Figura 28: Diagrama interno de nodos y de contenidos 2/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

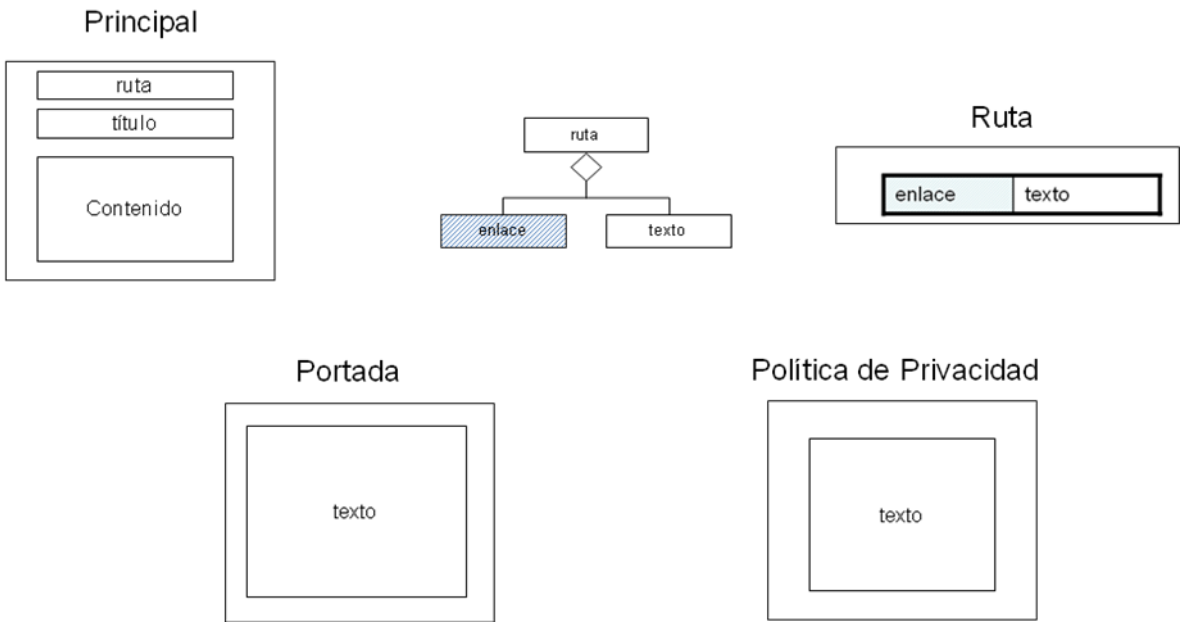


Figura 29: Diagrama interno de nodos y de contenidos 3/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

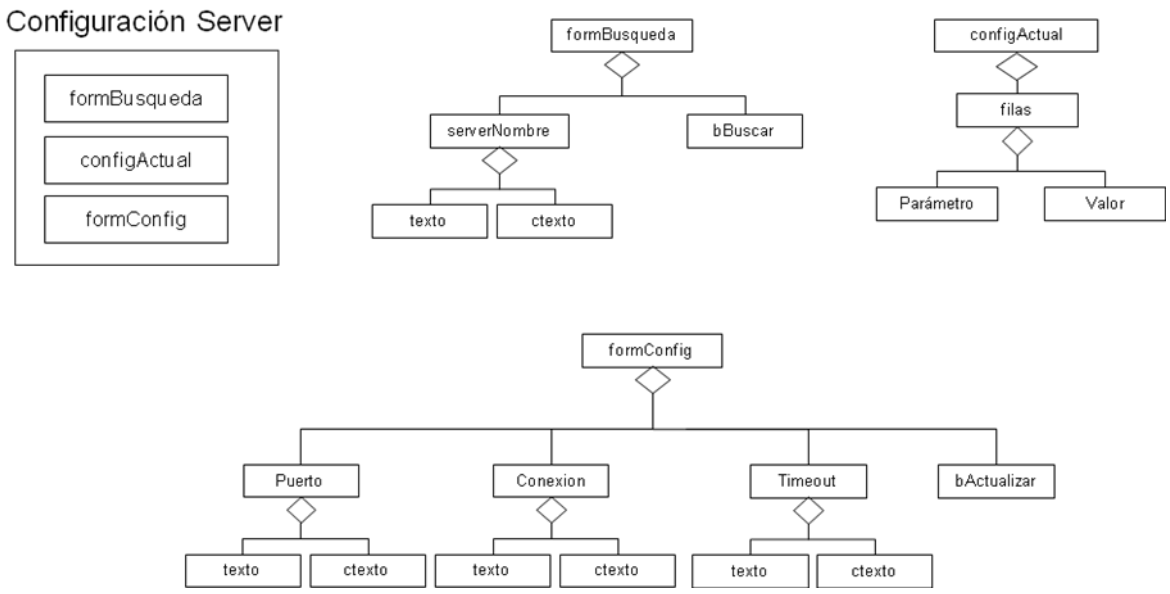


Figura 30: Diagrama interno de nodos y de contenidos 4/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

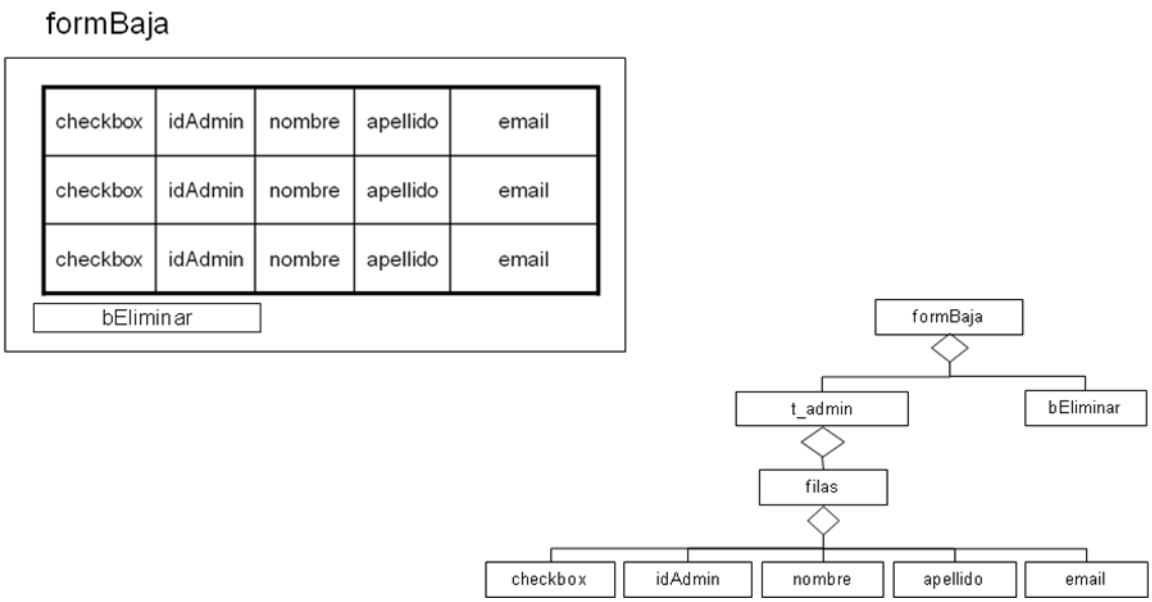


Figura 31: Diagrama interno de nodos y de contenidos 5/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

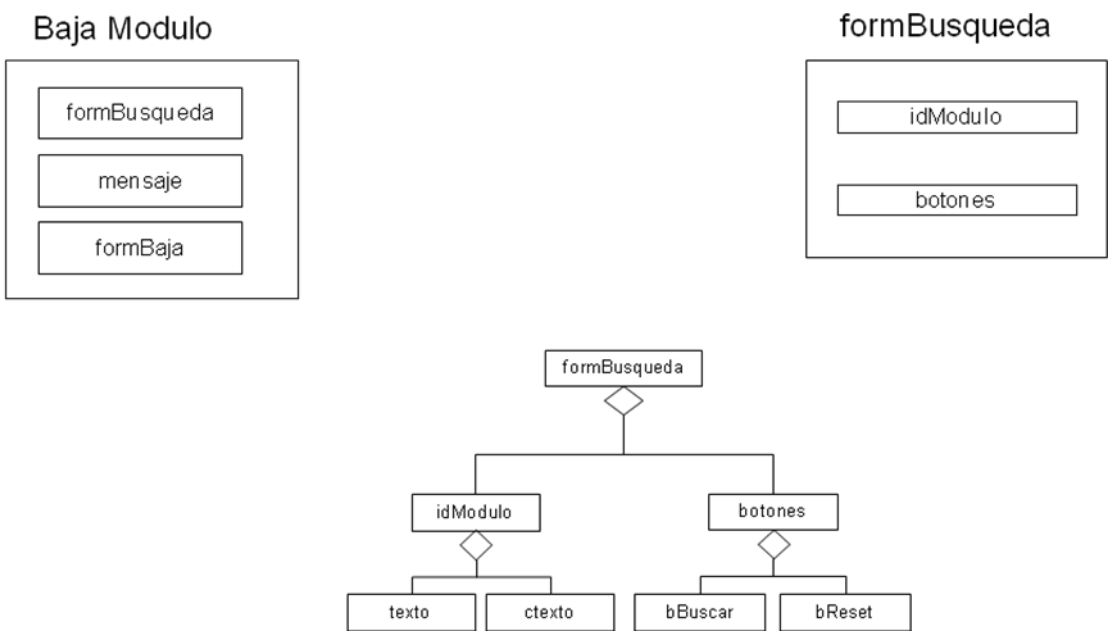


Figura 32: Diagrama interno de nodos y de contenidos 6/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

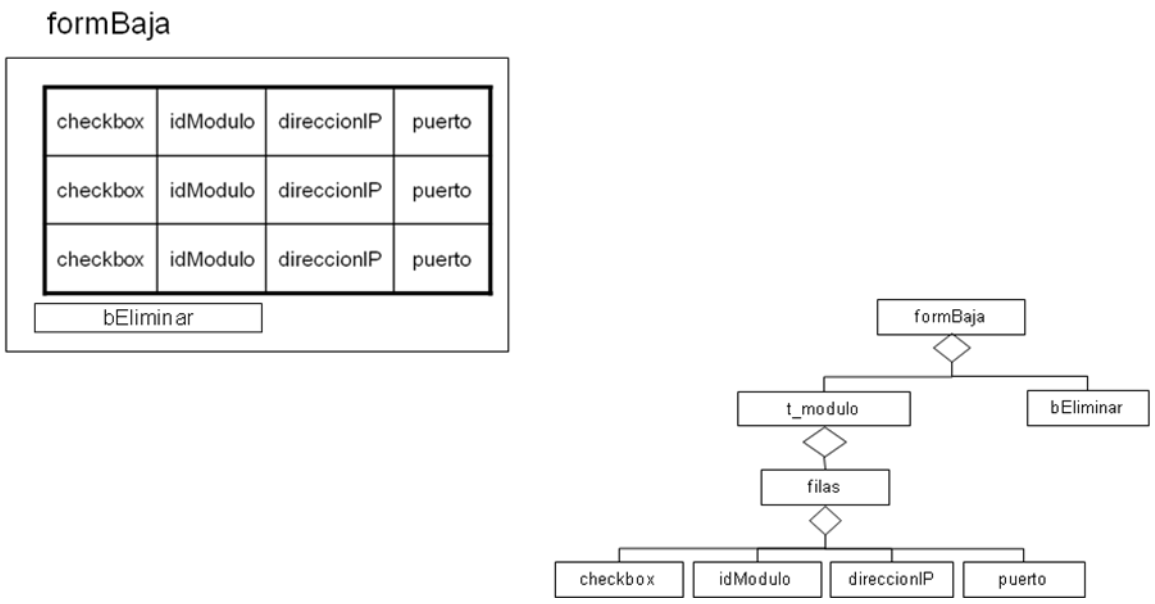


Figura 33: Diagrama interno de nodos y de contenidos 7/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

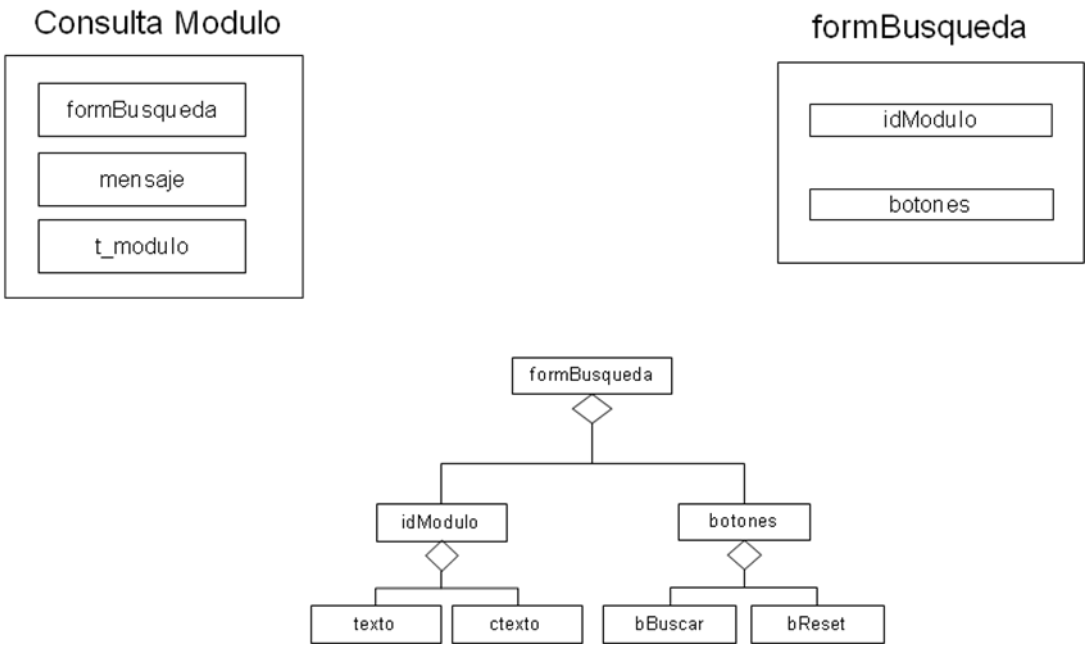


Figura 34: Diagrama interno de nodos y de contenidos 8/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

Figura 35: Diagrama interno de nodos y de contenidos 9/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

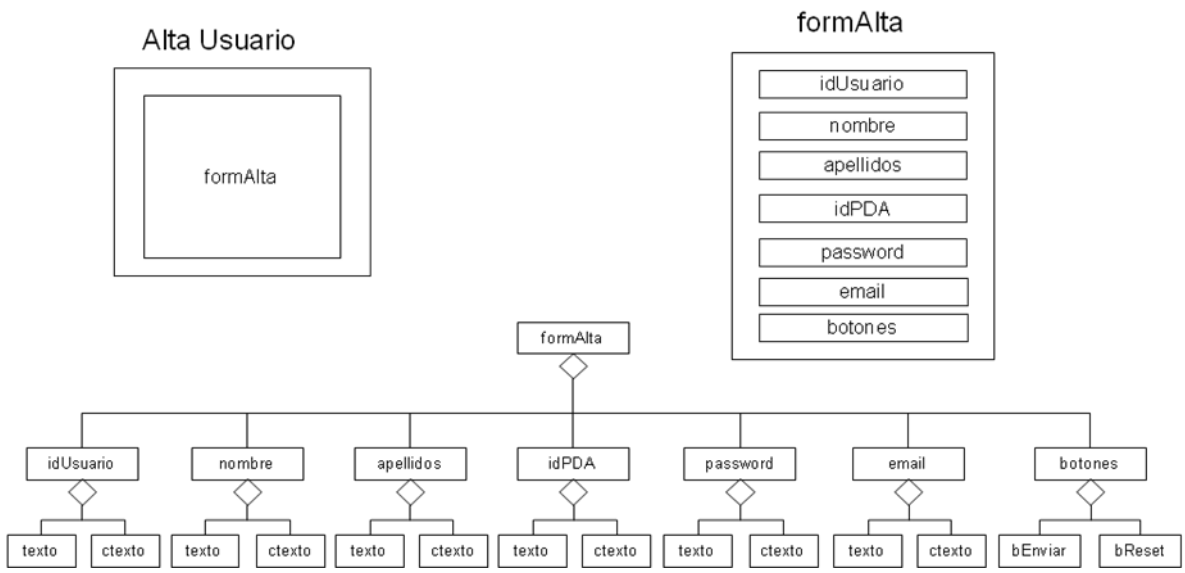


Figura 36: Diagrama interno de nodos y de contenidos 10/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

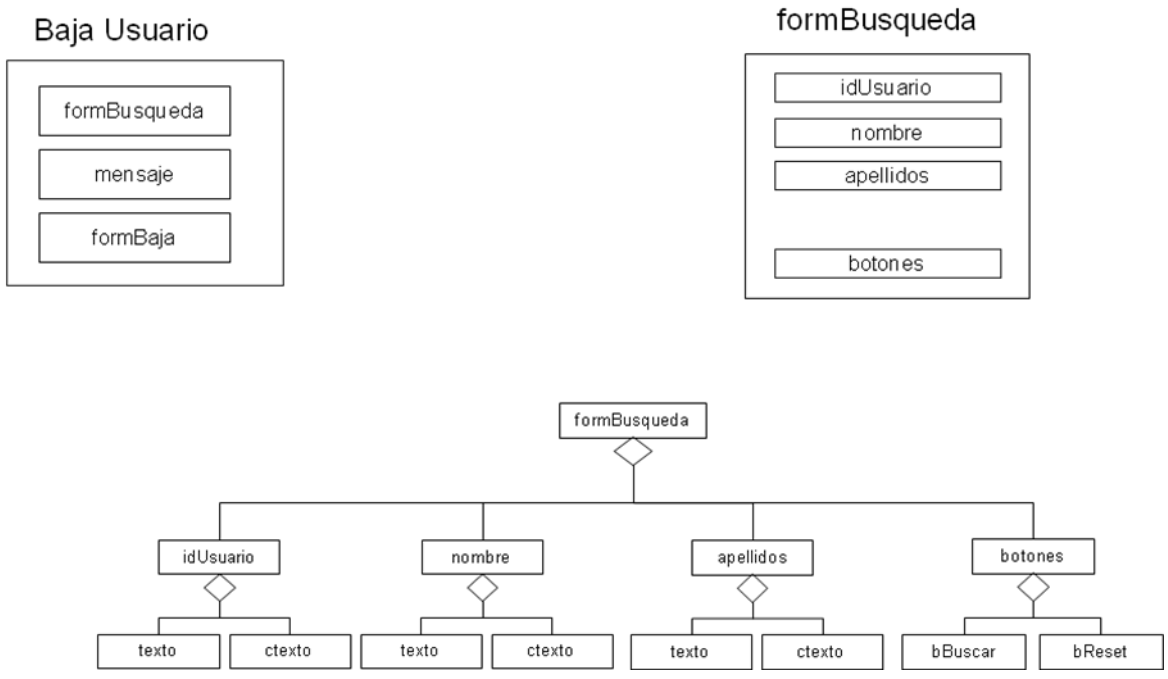


Figura 37: Diagrama interno de nodos y de contenidos 11/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

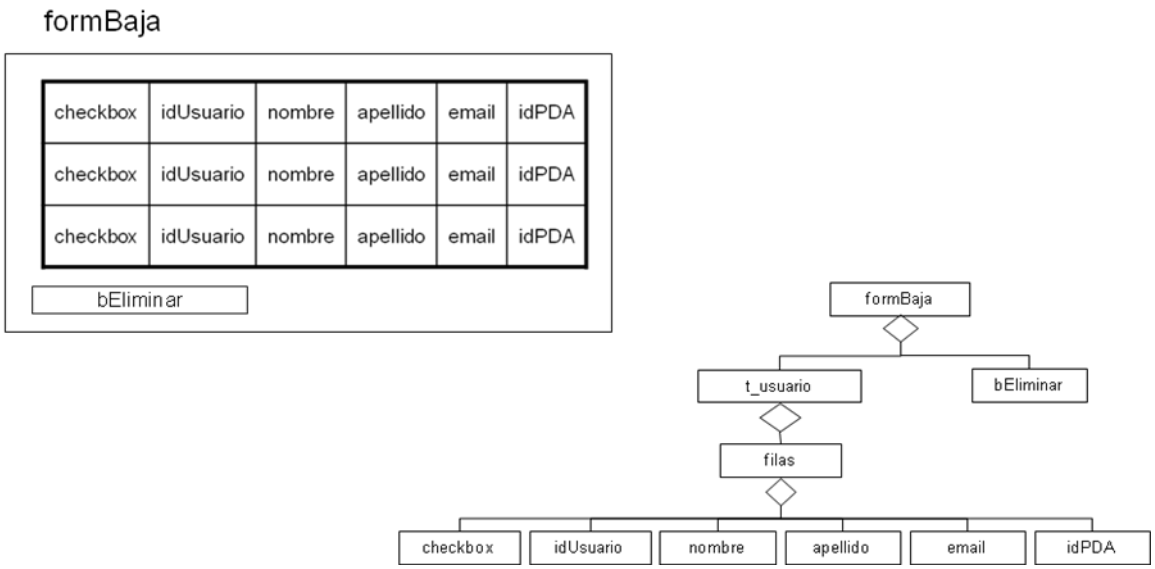


Figura 38: Diagrama interno de nodos y de contenidos 12/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

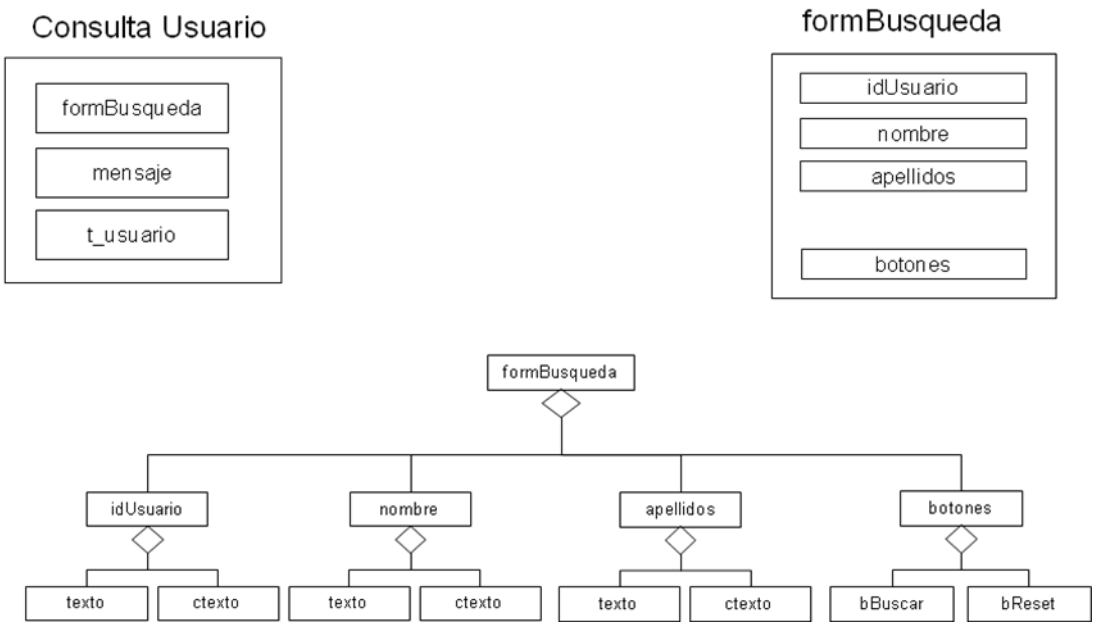


Figura 39: Diagrama interno de nodos y de contenidos 13/14 – Módulo de Administración.

Diagrama interno de nodos y de contenidos

t_usuario

idUsuario	nombre	apellido	email	idPDA
idUsuario	nombre	apellido	email	idPDA
idUsuario	nombre	apellido	email	idPDA

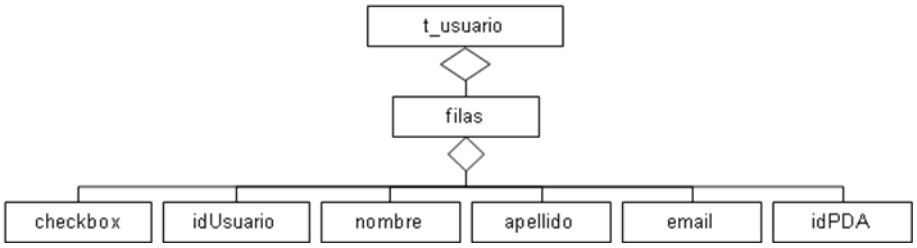


Figura 40: Diagrama interno de nodos y de contenidos 14/14 – Módulo de Administración.

Una vez mostrada la estructura de cada uno de los nodos o páginas que forman el sitio Web, vamos a ver cuál es su comportamiento.

El comportamiento de las páginas lo veremos reflejado en el catálogo de eventos, que consiste en la descripción de cada uno de los eventos que se pueden producir en los nodos del sitio Web.

La descripción de los eventos la realizaremos de forma semejante a la de la especificación de funcionalidades, es decir, utilizaremos una tabla para cada evento, en la que describiremos las propiedades del evento en cuestión. Los campos que contiene la tabla son:

- ID: identifica de forma única a un evento concreto.
- NAME: indica el nombre del evento.
- CONDITION: condición a cumplir para que se produzca el evento.
- OPERATION: operación a realizar cuando se activa el evento.
- RELATED FUNCTIONS: funciones relacionadas con el evento.
- RELATED EVENTS: eventos relacionados con el evento definido.

Catálogo de Eventos

ID:	E001
NAME:	Iniciar sesión
CONDITION:	Pulsar botón “Entrar" en el login
OPERATION:	Envía la información del administrador al sistema
RELATED FUNCTIONS:	F006
RELATED EVENTS:	

ID:	E002
NAME:	Desconectarse
CONDITION:	Pulsar botón “Salir" en la cabecera
OPERATION:	Cierra la sesión del administrador registrado
RELATED FUNCTIONS:	F007
RELATED EVENTS:	

ID:	E003
NAME:	Registrar
CONDITION:	Pulsar botón “Registrar" en el formulario de alta del administrador
OPERATION:	Enviar los datos del formulario
RELATED FUNCTIONS:	F001, F003
RELATED EVENTS:	

ID:	E004
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de alta del administrador
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F002
RELATED EVENTS:	

ID:	E005
NAME:	Consultar
CONDITION:	Pulsar botón “Buscar" en el formulario de baja del administrador
OPERATION:	Realiza una consulta en la tabla administradores en la BBDD.
RELATED FUNCTIONS:	F005
RELATED EVENTS:	

ID:	E006
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de baja del administrador
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F020
RELATED EVENTS:	

ID:	E007
NAME:	Eliminar Administrador
CONDITION:	Pulsar botón “Eliminar administrador" en el formulario de baja del administrador
OPERATION:	Elimina los administradores de la BBDD.
RELATED FUNCTIONS:	F004
RELATED EVENTS:	

ID:	E008
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de consulta del administrador
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F021
RELATED EVENTS:	

ID:	E009
NAME:	Consultar
CONDITION:	Pulsar botón “Buscar" en el formulario de consulta del administrador
OPERATION:	Realiza una consulta en la tabla administradores en la BBDD.
RELATED FUNCTIONS:	F005
RELATED EVENTS:	

ID:	E010
NAME:	Consulta
CONDITION:	Pulsar botón “Buscar" en el formulario de configuración del servidor.
OPERATION:	Realiza una consulta en la tabla configuración en la BBDD.
RELATED FUNCTIONS:	F018
RELATED EVENTS:	

ID:	E011
NAME:	Actualizar
CONDITION:	Pulsar botón “Actualizar configuración" en el formulario de configuración servidor.
OPERATION:	Realiza la actualización en la tabla configuración en la BBDD.
RELATED FUNCTIONS:	F019
RELATED EVENTS:	

ID:	E012
NAME:	Registrar
CONDITION:	Pulsar botón “Registrar" en el formulario de alta del usuario
OPERATION:	Enviar los datos del formulario
RELATED FUNCTIONS:	F008, F010
RELATED EVENTS:	

ID:	E013
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de alta del usuario
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F009
RELATED EVENTS:	

ID:	E014
NAME:	Consultar
CONDITION:	Pulsar botón “Buscar" en el formulario de baja del usuario
OPERATION:	Realiza una consulta en la tabla usuario en la BBDD.
RELATED FUNCTIONS:	F011
RELATED EVENTS:	

ID:	E015
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de baja del usuario
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F022
RELATED EVENTS:	

ID:	E016
NAME:	Eliminar Usuario
CONDITION:	Pulsar botón “Eliminar usuario" en el formulario de baja del usuario
OPERATION:	Elimina los usuario de la BBDD.
RELATED FUNCTIONS:	F011
RELATED EVENTS:	

ID:	E017
NAME:	Consultar
CONDITION:	Pulsar botón “Buscar" en el formulario de consulta del usuario
OPERATION:	Realiza una consulta en la tabla usuario en la BBDD.
RELATED FUNCTIONS:	F012
RELATED EVENTS:	

ID:	E018
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de consulta del usuario
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F021
RELATED EVENTS:	

ID:	E019
NAME:	Registrar
CONDITION:	Pulsar botón “Registrar" en el formulario de alta del modulo
OPERATION:	Enviar los datos del formulario
RELATED FUNCTIONS:	F013, F015
RELATED EVENTS:	

ID:	E020
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de alta del modulo
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F014
RELATED EVENTS:	

ID:	E021
NAME:	Consultar
CONDITION:	Pulsar botón “Buscar" en el formulario de baja del modulo
OPERATION:	Realiza una consulta en la tabla modulo en la BBDD.
RELATED FUNCTIONS:	F017
RELATED EVENTS:	

ID:	E022
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de baja del modulo
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F024
RELATED EVENTS:	

ID:	E023
NAME:	Eliminar Modulo
CONDITION:	Pulsar botón “Eliminar modulo" en el formulario de baja del modulo
OPERATION:	Elimina los modulo de la BBDD.
RELATED FUNCTIONS:	F016
RELATED EVENTS:	

ID:	E024
NAME:	Consultar
CONDITION:	Pulsar botón “Buscar" en el formulario de consulta del modulo
OPERATION:	Realiza una consulta en la tabla modulo en la BBDD.
RELATED FUNCTIONS:	F017
RELATED EVENTS:	

ID:	E025
NAME:	Resetear
CONDITION:	Pulsar botón “Borrar" en el formulario de consulta del modulo
OPERATION:	Borra los datos introducidos en el formulario
RELATED FUNCTIONS:	F025
RELATED EVENTS:	

Tabla 2: Catálogo de eventos – Módulo de Administración.

Para terminar, el último punto que nos falta por comentar está relacionado con la identificación de los usuarios y el tipo de acceso de cada uno ellos.

Los usuarios que pueden utilizar nuestro sitio Web son todos aquellos que sean administradores del módulo de comunicación, es decir, sólo existe un tipo de usuario dentro de módulo de administración, los administradores.

Tendríamos que tener en cuenta que cualquier persona podría acceder a la página de *login* de nuestro sitio Web, pero no al resto de las páginas, a no ser que inicie sesión. A consecuencia de esto, podemos observar que el único usuario que puede acceder a la totalidad de las páginas es el administrador del módulo de comunicación, el resto de usuarios sólo pueden acceder a la página de *login*.

A continuación, se muestra la tabla de acceso del sitio Web módulo de administración.

Tabla de Accesos

Núm. Página	***Nombre página***	***Núm. Usuario***	***Nombre Usuario***	***Permisos***
1	Login	1	Anónimo	Navegación
		2	Administrador	Navegación
2	Cerrarsesionadmin	1	Anónimo	Sin permisos
		2	Administrador	Navegación
3	Cabecera	1	Anónimo	Sin permisos
		2	Administrador	Navegación
4	Pie de página	1	Anónimo	Sin permisos
		2	Administrador	Navegación

5	Menu	1	Anónimo	Sin permisos
		2	Administrador	Navegación
6	Alta Usuario	1	Anónimo	Sin permisos
		2	Administrador	Navegación
7	Alta Modulo	1	Anónimo	Sin permisos
		2	Administrador	Navegación
8	Alta Admin	1	Anónimo	Sin permisos
		2	Administrador	Navegación
9	Política de Privacidad	1	Anónimo	Sin permisos
		2	Administrador	Navegación
10	Configuración Server	1	Anónimo	Sin permisos
		2	Administrador	Navegación
11	Baja Usuario	1	Anónimo	Sin permisos
		2	Administrador	Navegación
12	Baja Modulo	1	Anónimo	Sin permisos
		2	Administrador	Navegación
13	Baja Admin	1	Anónimo	Sin permisos
		2	Administrador	Navegación
14	Configuración Server Actual	1	Anónimo	Sin permisos
		2	Administrador	Navegación
15	Consulta Usuario	1	Anónimo	Sin permisos
		2	Administrador	Navegación
16	Consulta Modulo	1	Anónimo	Sin permisos
		2	Administrador	Navegación
17	Consulta Admin	1	Anónimo	Sin permisos
		2	Administrador	Navegación
18	Alta	1	Anónimo	Sin permisos
		2	Administrador	Navegación

19	Baja	1	Anónimo	Sin permisos
		2	Administrador	Navegación

Tabla 3: Tabla de acceso al módulo de administración.

Por último, tendríamos que destacar el empleo de hojas de estilo y de archivos de funciones en JavaScript.

Las hojas de estilo (archivo *.css) las hemos empleado para la personalización de la presentación de las páginas Web, y los ficheros de funciones JavaScript (archivos *.js) las hemos utilizado para implementar las funciones necesarias para la validación de los formularios.

4.3.2.4. Diseño del módulo del servidor

Antes de empezar con el diseño del módulo, vamos a realizar una breve descripción del mismo, donde comentaremos sus objetivos y los sub-módulos por los que está formado. El módulo tiene como finalidad principal iniciar el servidor del módulo de comunicación y procesar las peticiones entrantes.

Durante el inicio, el servidor obtiene sus parámetros de configuración de la base de datos. En caso de que no exista la configuración no se podrá iniciar el servidor y se mostrará un mensaje de error en la aplicación.

El servidor debe ser capaz de recibir y procesar las peticiones realizadas por la aplicación del usuario del dispositivo del móvil, además de enviar las respuestas generadas por el resto de módulos del sistema a la aplicación del usuario.

La operación de procesar las peticiones consiste en gestionar cada una de las peticiones, comprobando que módulo del sistema debe recibirla, enviársela a dicho módulo y esperar a que el módulo del sistema dé respuesta a la petición para poder reenviársela a la aplicación del usuario.

Hay que tener en cuenta que la comunicación entre ambos bloques del módulo de comunicación se produce bajo una conexión segura, con lo que aseguramos la confidencialidad de los datos transmitidos por la red.

Hay que destacar que el servidor puede recibir y procesar de forma simultánea varias peticiones, ya que está diseñado como una aplicación multiproceso.

El diseño del módulo se ha basado en la realización de un diagrama de clases, como se ha llevado a cabo en el modelado de los módulos anteriores. El diagrama de clases diseñado para el módulo del servidor está formado por diez clases, relacionadas entre sí, incluso existe una estructura de herencia.

La estructura de herencia presente en el diagrama se debe a la abstracción que se puede realizar de los distintos tipos de mensajes a tratar, definiendo el mensaje más simple como la clase base, Mensaje, de la que heredan el resto de mensajes, que son: MensajePeticion y MensajeServer. Esta estructura de herencia es igual que la que hemos descrito en el módulo de comunicación desplegado en el dispositivo móvil (apartado 4.3.1.).

Las diez clases existentes en el diagrama de clase son las siguientes:

- Clase Principal: inicia el servidor y acepta las peticiones de conexión por parte de la aplicación del usuario, creando un hilo de ejecución para cada una de las peticiones.

- Clase Config: contiene los atributos y métodos necesarios para la carga de la configuración del servidor y la búsqueda de las configuraciones de los módulos de peticiones del sistema.

- Clase ConfigServer: contiene la información relativa a la configuración de cada módulo que componen el sistema.

- Clase Servidor: se encarga de procesar la petición recibida y la reenvía al módulo del sistema que corresponda, para que éste la procese y nos envíe la respuesta, y el módulo de comunicación a su vez la reenvíe a la aplicación del usuario del dispositivo móvil.

- Clase CifradorRSA: contiene todos los atributos y métodos necesarios para la gestión del cifrado del mensaje.

- Clase MensajeCifrado: es la representación del mensaje cifrado, que contiene los datos del mensaje cifrado (cifrado con AES), el vector de inicialización y la clave (cifrado con el sistema de cifrado RSA) del sistema de cifrado simetrico AES.

- Clase Mensaje: simboliza la abstracción más simple del mensaje. Se encarga de comprobar el formato del mensaje.

- Clase MensajePeticion: es la abstracción del mensaje de una petición. Tiene como objetivo comprobar las etiquetas del mensaje de petición.

- Clase MensajeServer: es la representación del mensaje de petición que recibe el servidor del módulo de comunicación. Su finalidad es comprobar el formato del mensaje.

- Clase Usuario: contiene la información que se refiere al usuario que ha realizado la petición.

Las clases principales del módulo son la clase Principal y la clase Servidor.

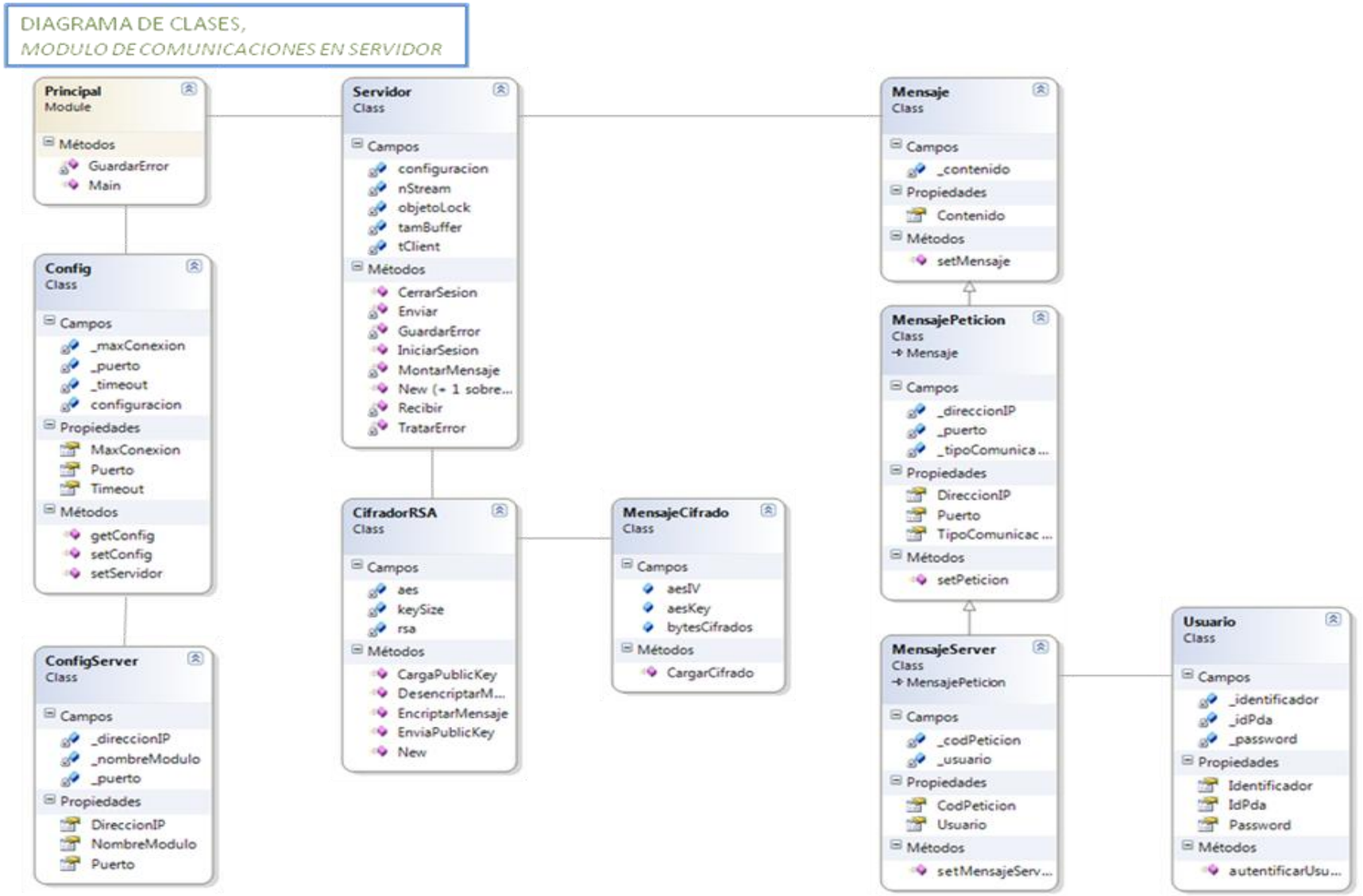


Figura 41: Diagrama de clases – Módulo del Servidor.

Debido a que la implementación de las clases es la misma, omitiremos la explicación detalla de aquellas que hemos explicado anteriormente en el diseño del módulo de comunicación implementado en el dispositivo móvil, aunque recordaremos cuáles eran sus objetivos principales.

Las clases afectadas son:

- Las clases pertenecientes a la estructura de herencia, que son la clase Mensaje, la clase MensajePeticion y la clase MensajeServer. Estas clases tenían como finalidad la carga de los mensajes en su respectiva clase y la comprobación del formato del mensaje.

- Las clases CifradorRSA y MensajeCifrado. Estas clases eran las encargadas del cifrado y el descifrado de los mensajes que se transmitían entre los dos bloques que constituyen el módulo de comunicación.

- La clase Usuario. Esta clase recogía toda la información perteneciente al usuario que realizaba la petición. Aunque, tenemos que decir que la implementación en el módulo del servidor es distinta a la anterior, ya que se ha agregado la función `AutentificarUsuario()` a la clase.

  Esta función se encarga de autenticar el usuario de la petición, ya que sólo los usuarios registrados en el sistema pueden utilizar sus servicios. No tiene ningún parámetro, ni de entrada ni de salida. La función devuelve True siempre que la autenticación del usuario haya sido satisfactoria y devuelve False en el caso contrario.

  La implementación de la función es muy parecida a la función de autenticación de un administrador del módulo de administración, por este motivo no se expondrá el pseudocódigo.

Después, de realizar un breve recordatorio de las clases explicadas anteriormente, realizaremos una explicación detalla del resto de clases del módulo del servidor.

4.3.2.4.1. Descripción de las clases Config y ConfigServer

Ambas clases están relacionadas con la carga de la configuración del servidor y la obtención de las configuraciones de los módulos del sistema (los que procesarán las peticiones del usuario). Empezaremos explicando la clase ConfigServer antes que la clase Config, para que esta última sea comprendida con más facilidad.

La clase ConfigServer únicamente contendrá la información relativa a la configuración de cada uno de los módulos que componen el sistema. Esta clase sólo contiene en su definición una serie atributos que contendrán los datos de los módulos.

Los atributos por los que está compuesta la clase ConfigServer son:

- `_nombreModulo`: contiene el nombre que identifica a un módulo de forma unívoca.

- `_direccionIP`: contiene la dirección IP donde se está ejecutando el módulo del sistema.

- `_puerto`: contiene el puerto en el que está escuchando el módulo del sistema.

La clase Config tiene definidos todos los atributos, métodos y funciones necesarias para la carga de la configuración del servidor y la carga de las configuraciones del resto de módulos del sistema. Debe importar la librería *System.Data.SqlClient* de .Net para poder tener acceso al gestor de base de datos.

Los atributos definidos en la clase Config son:

- `_puerto`: contiene el valor del puerto en el que va escuchar el servidor. Este valor se carga de la configuración del servidor que se encuentra en la base de datos.

- `_maxConexion`: contiene el número de conexiones máximas que puede ejecutar el servidor de manera simultánea. Este valor se carga de la configuración del servidor que se encuentra en la base de datos.

- `_timeout`: contiene el valor del periodo de tiempo durante el cual el servidor esperará para recibir datos tras iniciarse una operación de lectura. Este valor se carga de la configuración del servidor que se encuentra en la base de datos.
- `Configuración`: vector de objetos de la clase ConfigServer. Este vector estará formado por las configuraciones del resto de módulos del sistema.

Los métodos y funciones implementadas en la clase Config son:

- `setServidor()`: esta función establece la configuración que va a ser usada por el servidor, la configuración se carga de la tabla configuración de la BD.

La función recibe como parámetro de entrada el nombre del servidor, del que se va a cargar la configuración.

Esta función devuelve True si la carga de la configuración ha sido correcta, y devuelve False en caso contrario.

- `setConfig()`: este método carga la configuración de los módulos del sistema que recibirán las peticiones de los usuarios.

  El método no tiene parámetros de salida, ni tampoco de entrada.

- `getConfig()`: esta función es la encargada de buscar la configuración de un módulo (los servidores que reciben la peticiones de los usuarios) en concreto.

  La función recibe como parámetro de entrada el nombre del módulo que se debe buscar. Como parámetro de salida tiene un objeto de la clase ConfigServer, en la que se almacenará la configuración del módulo solicitado, si este existe.

  Esta función devuelve True cuando la carga de la configuración ha sido satisfactoria, si no devolverá False.

4.3.2.4.2. Descripción de la clase Principal

La clase Principal es la más importante del módulo del servidor, ya que inicia el servidor y acepta las peticiones de conexión por parte de la aplicación del usuario, creando un hilo de ejecución para cada una de las peticiones.

Para que esta clase pueda funcionar correctamente necesitaría importar las librerías *System.Net* y *System.Net.Sockets* de .Net para tratar las conexiones de red y la librería *System.Threading* de .Net para realizar las operaciones de multiproceso del servidor (poder procesar de manera simultánea varias peticiones).

Esta clase no posee ningún atributo, pero sí tiene definidos dos métodos. Estos métodos son:

- `GuardarError()`. Método que se encarga de guardar en el fichero de registro de errores, “servidor.log”, las características de un error que se ha producido durante el inicio del servidor.

El método recibe como parámetros de entrada la cadena de error personalizada y la característica de la excepción que se ha producido.

- `Main()`. Este es el método principal de la clase, y de la aplicación, ya que es el método que se invoca de forma automática cada vez que se inicia el servidor.

  Es el encargado de iniciar el servidor y de crear los hilos que procesarán la petición recibida. Este método no recibe ningún parámetro de entrada ni de salida. Debido a la importancia del método, a continuación, se muestra su pseudocódigo:

```
método Main()
{
' nombre del servidor en la BD
nomServidor = "Servidor001"
' cargamos la configuración del servidor desde la BD
si CargarConfiguracion(nombServidor,config) = TRUE entonces
    ' la carga de la configuración ha sido correcta
    ' cargamos la configuración  del resto de módulos del sistema
    CargarConfigModulo()
    ' establecemos las características del servidor con la configuración
    servidor.Configurar(direccionIP,config.puerto)
    ' arrancamos el servidor
    servidor.Iniciar(config.maxConexion)
    ' creamos un bucle infinito, para poder aceptar más de una petición
    mientras TRUE entonces
            ' esperamos a recibir peticiones y las aceptamos
            cliente = servidor.AceptarConexion()
            ' creamos un objeto servidor, para la sesión de la petición
            server = CrearServer(cliente)
            ' creamos el hilo, para ejecutar la petición
            hilo = CrearHilo(server.Iniciar())
            ' iniciamos el hilo
            hilo.Iniciar()
sino
    ' la carga de la configuración ha sido incorrecta. ERROR
    ' guardamos el error en el fichero
    GuardarError()
}
```

4.3.2.4.3. Descripción de la clase Servidor

La clase Servidor procesa la petición recibida y la reenvía al módulo del sistema que corresponda, para que éste la procese y envíe la respuesta a la petición, y el módulo de comunicación reenviársela a la aplicación del usuario del dispositivo móvil. Esta clase debe importar las librerías *System.Net* y *System.Net.Sockets* de .Net para poder tratar las conexiones de red.

Los atributos definidos en la clase Servidor son:

- `objetoLock`: atributo definido como compartido dentro de la clase. Con esto permitimos que todas la instancias de la clase Servidor compartan el atributo, que utilizaremos como un semáforo a la hora de acceder al fichero de registro de errores, "servidor.log".

  Lo que pretendemos con la declaración del atributo es bloquear el acceso al archivo cuando está en uso y desbloquearlo cuando es liberado.

- `Configuración`: contiene la configuración del servidor y de los módulos del sistema. Es un objeto de la clase Config.

- `nStream`: proporciona los métodos necesarios para el tratamiento de la conexión establecida con el cliente.

- `tClient`: contiene información relativa a la conexión establecida con el cliente a través de una red TCP.

Los métodos y funciones declaradas en la clase Servidor son las siguientes:

- `New()`: es un método que está sobrecargado, ya que se encuentra declarado dos veces, pero con signaturas diferentes. La primera declaración del método recibe como parámetros la configuración del servidor y el objeto de la TcpClient que contiene la información sobre la conexión del cliente. La segunda declaración del método sólo recibe como parámetro el objeto de la clase TcpClient que contiene los datos de la conexión del cliente.

- `Recibir()`: la función tiene como misión la recepción de los datos por parte del servidor del sistema. Devuelve los datos recibidos.

- `Enviar()`: método que envía los datos de la petición al servidor. Recibe como parámetros de entrada los datos a enviar.

- `CerrarSesion()`: método encargado del cierre de la sesión, es decir, cierra la conexión existente entre un módulo y el servidor.

- `TratarError()`: método encargado del envío de errores, producidos en el servidor durante la sesión del usuario. El método recibe como parámetro de entrada la cadena personalizada de error que hay que enviar al usuario.

- `MontarMensaje()`: esta función construye el mensaje que va ser enviado al usuario, una vez que ya ha sido procesada su petición.

  La función recibe como parámetro de entrada los datos enviados por el servidor de peticiones. Devuelve los bytes que van a ser enviados al usuario.

- `GuardarError()`: este método guarda en el fichero de registro de errores, "servidor.log", las características de un error que se ha producido en el servidor. El método recibe como parámetros de entrada la cadena personalizada de error y las características de la excepción que ha producido.

  A continuación, se muestra el pseudocódigo de esta función, ya que es interesante que veamos cómo se realiza el bloqueo y el desbloqueo del recurso (fichero "servidor.log"), para que no existan errores del tipo "Error al acceder al fichero porque el fichero se encuentra en uso." o parecidos.

  Pseudocódigo:

```
método GuardarError(cadena, excepcion)
{
' bloquear el fichero, aunque verdad lo que se bloquea es una sección del código.
SyncLock objetoLock
    ' abrimos el fichero en modo escritura y escribimos al final
fichero = AbrirFich("servidor.log",Append)
' escribimos la hora, la cadena de error y la excepción
fichero = Ecribir(FechaActual)
fichero = Ecribir(cadena)
fichero = Ecribir(excepcion)
' cerramos el fichero
fichero.Cerrar()
' desbloquea la sección de código
End SynsLock
}
```

- `IniciarSesion()`: este método realiza el inicio de una sesión y engloba todas las acciones de la sesión de un cliente. El método no tiene ningún parámetro de entrada, ni de salida.

  Debido a la importancia de este método, vamos a mostrar a continuación el pseudocódigo del método.

Pseudocódigo:

```
método IniciarSesion()
{
‘ creamos un sistema de cifrado RSA
cifradorRSA = CrearCifradorRSA()
‘ obtenemos la clave pública del cifrador y se la enviamos al usuario
clavePublica = cifradorRSA.EnviarPublicKey()
Enviar(clavePublica)
‘ esperamos a recibir los datos de la peticion cifrados
bDatos = Recibir()
‘ cargamos los datos en MensajeCifrado y los descifframos
mensajeCifrado = CrearMensajeCifrado()
mensajeCifrado.CargarCifrado(bDatos)
sDatos = cifradorRSA.Descifrar(mensajeCifrado)
‘ cargamos los datos en la clase MensajeServer, para comprobar el formato.
mensaje = CrearMensajeServer()
si mensaje.CargarMensajeServer(sDatos) = TRUE entonces
    ‘ comprobamos el usuario
    usuario = mensaje.usuario
    si usuario.AutenticarUsuario() = TRUE entonces
            ‘ cargamos la configuración del modulo de la peticion
            config = CrearConfigServer()
            si config.GetConfig(mensaje.codPeticion) entonces
                    ‘ establecemos la conexión con el módulo
                    tcpServer = CrearTcpClient()
                    tcpServer.Conectar(config.direccion,config.puerto)
                    ‘ enviamos la petición del usuario
                    Enviar(bDatos)
                    ‘ esperamos la respuesta del módulo a la petición
                    moduloRespuesta = Recibir()
                    ‘ cerramos la conexión
                    tcpServer.Cerrar()
                    ‘ enviamos la respuesta al usuario
                    Enviar(MontarMensaje(moduloRespuesta))
            sino
                    ‘ se ha producido un error al buscar la configuración
                    TratarError(“Configuracion del módulo no existe”)
    sino
            ‘ se ha producido un error al autenticar usuario
            TrataError(“Usuario incorrecto.”)
sino
    ‘ se ha producido un error en el formato del mensaje
    TratarError(“Formato del mensaje incorreto.”)
‘ cerramos la conexión
CerrarSesion()
}
```

Con esto concluimos la explicación del diseño y de la implementación del módulo del servidor, y como consecuencia también acabamos con la explicación del módulo de comunicación desplegado en el servidor del sistema.

CAPÍTULO 5. RESULTADOS

5.1. Resultados generales

El resultado principal de este PFC ha sido el desarrollo de un módulo de comunicaciones que se integrará dentro del sistema de gestión turística descrito en el apartado 2.1.5.

El módulo de comunicaciones que se ha desarrollado en el proyecto se puede dividir en dos bloques distintos, de acuerdo a la siguiente arquitectura:

Bloque 1. Módulo de comunicación desplegado en el dispositivo móvil. Este bloque es capaz de transmitir las peticiones realizadas por la aplicación del usuario al servidor del sistema, es decir, sirve de enlace entre la aplicación del usuario en la PDA y el resto de módulos del sistema.

Es una librería DLL, que consta de una serie de métodos y funciones con las que se realizará el tratamiento de las peticiones. Esta librería deberá ser importada por la aplicación de usuario para poder realizar el envío de las peticiones al servidor del sistema.

Bloque 2. Módulo de comunicación desplegado en el servidor del sistema. Este módulo tiene la capacidad de recibir todas las peticiones realizadas por el otro bloque del módulo de comunicación, procesa dichas peticiones para que sean ejecutadas por el resto de módulos del sistema, y responde al usuario con la respuesta obtenida de los servidores. Además, se encarga de realizar la gestión y la administración de los recursos del sistema.

El módulo de servidor está formado por:

- Módulo principal, ServidorPC: es el módulo principal que será capaz de recibir y procesar las peticiones realizadas por la aplicación del usuario del dispositivo del móvil.
- Módulo de administración: se encarga de la gestión de todos los recursos que interaccionan con el módulo de comunicación (la gestión de usuarios, administradores y módulos del sistema).
- Módulo de creación de la base de datos: tiene el objetivo de crear una base de datos con las configuraciones iníciales del servidor.

- Base de datos: almacena toda la información relativa a los usuarios, los administradores y la configuración de los módulos del sistema, además de la configuración del servidor.

Como ya se comentó, el objetivo principal del proyecto era, *permitir una comunicación entre todos los módulos que componen el sistema*, pero fundamentalmente entre la aplicación de usuario, que reside en el dispositivo móvil, y los módulos que reciben las peticiones.

Otros objetivos secundarios, pero no menos importantes eran:

- La comunicación entre la aplicación de usuario y los módulos que reciben las peticiones debe ser fluida.
- La comunicación entre la aplicación de usuario y el módulo de comunicación de la parte del servidor del sistema debe de ser una conexión segura, debido a que se intercambia información privada del cliente.
- Soportar el crecimiento continuo de trabajo de la manera más fluida.
- El módulo debe ser capaz de tolerar fallos sin alterar el funcionamiento del sistema.

Comprobaremos, por lo tanto, si los objetivos marcados al principio del proyecto se han cumplido:

Para poder cumplir el objetivo principal del proyecto fin de carrera (ser capaz de permitir una comunicación entre todos los módulos que componen el sistema de gestión de información turística) hemos tenido que realizar el diseño y la implementación de un módulo de comunicación que permita a la aplicación del usuario utilizar los servicios ofrecidos por el sistema. El diseño de la arquitectura empleada en la consecución de este objetivo está especificado al comienzo del capítulo.

Conseguido este objetivo vemos que todos los requisitos secundarios que tenía que cumplir han sido satisfechos con el diseño e implementación del módulo de comunicación, como son: la escalabilidad, el nivel de seguridad de la conexión y la robustez.

Para conseguir el objetivo de escalabilidad se lleva a cabo la creación del módulo servidor como una aplicación multiproceso, es decir, el servidor puede ejecutar peticiones de manera simultánea, aumentando el número de estas hasta el nivel de procesamiento del equipo.

El requisito de la seguridad en la conexión se realiza mediante el cifrado del mensaje con un sistema de cifrado AES y la clave de cifrado de éste con un sistema de cifrado de RSA, manteniendo así la confidencialidad de los datos del usuario.

La robustez se ha logrado considerando todos los errores posibles y tratándolos para que el sistema no sufra ninguna modificación en su funcionamiento.

5.2. Pruebas

Para verificar que el objetivo se ha cumplido, no solo de forma teórica sino también de forma práctica, se ha sometido a una serie de pruebas a ambos sub-módulos del módulo de comunicación.

5.2.1. Pruebas en el servidor

La batería de pruebas consistió en realizar una aplicación multiproceso que lanzará un número determinado de clientes contra el módulo de comunicación, y la creación de otra pequeña aplicación que actuará como los módulos que procesarán las peticiones. En concreto, en la batería de pruebas se realizaron ensayos con cien, doscientas y trescientas peticiones de clientes, en los que el módulo de comunicación tenía que reenviar a la aplicación que actuaba como módulo de peticiones y esperar su respuesta para poder reenviárselas a los distintos clientes. La comunicación entre el módulo servidor y la aplicación cliente se realizó cifrada.

Esta batería de pruebas tuvo un gran éxito, aunque surgieron algunos errores que fueron solucionados posteriormente, como era el acceso al fichero de registro de errores cuando dos peticiones intentaban acceder a él.

Otras pruebas llevadas a cabo para comprobar la robustez de la aplicación tenían como finalidad provocar todos los tipos de errores posibles y comprobar que el funcionamiento del módulo de comunicación y tratamiento de los errores es el correcto.

5.2.2. Pruebas en el dispositivo móvil (emulador)

Las pruebas que se han llevado a cabo para el sub-módulo desplegado en el dispositivo móvil se han realizado con la utilización de un emulador, ya que no se disponía de este dispositivo durante el desarrollo del proyecto. Estas pruebas han consistido en la creación de una pequeña aplicación de ejemplo, que simulaba a la aplicación del cliente del dispositivo móvil, con la que realizamos el envío de una petición al sistema de gestión de información turística, utilizando el módulo de comunicación.

Debido a limitaciones en el simulador, en concreto la imposibilidad de configurar la red del emulador del dispositivo móvil, no ha sido posible enviar la petición al otro módulo de comunicación del servidor, aunque tanto el cifrado del mensaje como la comprobación del formato del mensaje de petición funcionan correctamente, ya que el código de la librería es muy similar al del módulo servidor.

En un futuro se realizarán las pruebas oportunas con un dispositivo móvil real, comprobando si la comunicación entre ambos módulos del sistema se lleva a cabo con éxito.

CAPÍTULO 6. CONCLUSIONES

6.1. Conclusiones generales

Como primera conclusión de este PFC podemos decir que se han cumplido satisfactoriamente los objetivos que se marcaron al principio del proyecto. En concreto se han cumplido los siguientes objetivos:

- El módulo de comunicación sirve de enlace entre la aplicación de usuario que reside en el dispositivo móvil y el resto de módulos del sistema que procesan las peticiones.
- La conexión entre el módulo de comunicación desplegado en la PDA y el módulo servidor está cifrada mediante los sistemas de cifrado AES y RSA.
- Posee tolerancia a fallos durante el procesamiento de las peticiones, es decir, el módulo debe seguir funcionando correctamente en el caso de que se produzca un error.
- Es capaz de gestionar un crecimiento continuo de la carga de trabajo de manera fluida.

Acerca del desarrollo del proyecto podemos decir que se ha realizado de forma progresiva. Lo primero que se realizó fue la definición de todos los requisitos que tenía que cumplir el módulo de comunicación, para que cumpliera los objetivos y pudiera ser desplegado dentro del sistema de gestión de información turística. A continuación, se realizó el diseño del módulo de comunicación basado en los objetivos especificados anteriormente. En la fase de diseño se dividió en dos bloques, debido a los requisitos que tenía que cumplir (sirve de enlace entre la aplicación del usuario del dispositivo móvil y el servidor de peticiones).

Una vez terminadas estas tareas, se procedió a la implementación de los sub-módulos que formaban el módulo de comunicación. Primero se implementó el módulo servidor, debido fundamentalmente a la importancia que tiene éste dentro del módulo de comunicación, ya que recibe las peticiones realizadas por el cliente y las trasmite para obtener la respuesta que devolverá al cliente. Dentro de este módulo a su vez se tuvo que realizar el diseño de otras dos aplicaciones para poder gestionar el servidor de la mejor forma posible. Una de las aplicaciones consistía en la realización de un sitio web que diera soporte a la configuración del servidor y la gestión de los usuarios, administradores y módulos del sistema; la otra aplicación tenía como finalidad la elaboración de la base de datos que utilizaría el servidor.

Después de implementar el sub-módulo, se realiza la batería de pruebas oportunas para comprobar que el funcionamiento del módulo de comunicación desplegado en el servidor del sistema es el correcto.

Terminada la implementación y las pruebas del primer sub-módulo, se procede a la implementación del segundo sub-módulo del módulo de comunicación. La implementación de este módulo fue mucho más sencilla que la anterior, ya que gran parte del código empleado para el desarrollo del primer módulo es reutilizado para la implementación del segundo sub-módulo, aunque haciendo unas pequeñas variaciones debido a que éste será desplegado en el dispositivo móvil (no tiene las mismas capacidades de procesamiento que un ordenador). Posteriormente, se llevaron a cabo las pruebas con el emulador del dispositivo, que fueron bastante satisfactorias.

Por último, se procedió a la elaboración de las pruebas generales, en la que se intentaba comprobar que ambos sub-módulos podían comunicarse y procesar las peticiones de los clientes, pero esto no fue posible del todo, ya que el emulador no aceptaba conexiones de red, solo permitía acceso local dentro del mismo emulador.

Atendiendo al tema de los problemas que se han encontrado en el desarrollo del proyecto se puede decir que han existido bastantes, como se ha podido ir viendo a lo largo de la memoria, sobre todo a la hora del cifrado del mensaje y a la hora de acceder al fichero de registro de errores, “servidor.log”, del servidor. Aunque, finalmente se han ido resolviendo todos ellos.

En conclusión, podemos decir que el proyecto desarrollado, el módulo de comunicación, se ha llevado a cabo satisfactoriamente con lo que está preparado para su despliegue en el sistema global.

6.2. Ventajas e inconvenientes del diseño realizado

En primer lugar destacamos la utilización de la arquitectura cliente-servidor. Esta arquitectura nos da la capacidad de separar responsabilidades y gestionar la información de forma centralizada, debido que el procesamiento está repartido entre los clientes y los servidores, lo que contribuye a la realización de un diseño más claro de la aplicación.

La ventaja principal que encontramos en el módulo de comunicación desarrollado en el proyecto, es que nos permite establecer una conexión segura (porque los mensajes que circulan por la red están cifrados) entre la aplicación de usuario del dispositivo móvil y los servidores que procesarán las peticiones.

No podemos olvidar la ventaja de tener centralizadas todas las comunicaciones que se producen en el sistema de gestión de información turística, ayudando en gran medida a que la aplicación del usuario no tenga que realizar cada una de las peticiones a los distintos servidores directamente, sino que es el módulo de comunicación quien se encarga de ello.

Esta última ventaja puede actuar como un arma de doble filo, debido a que todas las comunicaciones estén centralizadas, liberando así de trabajo a la aplicación del usuario, pero provoca que el módulo de comunicación se pueda colapsar, debido al alto tráfico que tiene que soportar, a no ser que el servidor del módulo de comunicación se encuentre desplegado en un equipo suficientemente potente para procesar tal cantidad de peticiones. Para terminar, podemos decir que no existen grandes inconvenientes, ya que el único expuesto puede ser solucionado con la utilización de un equipo con mayor capacidad, si esto fuera necesario.

Otra posibilidad sería desplegar distintos módulos de este tipo de modo que se dividiesen los clientes a los que dé soporte cada uno de ellos, e incluso si fuera necesario duplicando el resto de módulos (módulo de mapas, módulo de monumentos…) del sistema.

CAPÍTULO 7.
FUTURAS LÍNEAS DE TRABAJO E INVESTIGACIÓN

En este capítulo se van a tratar las líneas de investigación y ampliaciones futuras que se podrían aplicar al módulo de comunicación desarrollado en el proyecto para aumentar las funcionalidades que puede ofrecer, además de mejorar las ya existentes. Estas líneas de investigación y ampliaciones van desde el mejoramiento del nivel de seguridad hasta la optimización de las aplicaciones.

Sobre todo tendríamos que tener en cuenta todas las líneas de investigación y trabajo futuras relacionadas con el nivel de seguridad de la aplicación, ya que el proyecto desarrollado es un módulo de comunicación, que será mucho más susceptible a sufrir ataques que otros tipos de aplicaciones.

7.1. Nivel de seguridad

Dentro de este apartado, tendríamos que decir que la seguridad de un sistema informático es relativa, ya que la seguridad es el resultado del equilibrio entre el riesgo y las medidas establecidas para paliarlo.

Una de las posibles líneas de trabajo que podríamos realizar dentro del proyecto relacionado con el nivel de seguridad sería que las claves, tanto de clientes como de los administradores, que se encuentran en la base de datos del servidor, estuvieran cifradas mediante algún sistema de cifrado, ya que algún atacante podría intentar acceder a la base de datos para obtener todas la contraseñas de los usuarios.

Un ejemplo de sistema de cifrado de las claves de la base de datos podía ser el empleo de una función irreversible. Este tipo de cifrado consiste en cifrar un mensaje mediante el uso de un algoritmo no invertible, es decir, el mensaje se cifra pero no puede ser descifrado. Por este motivo, sería una buena elección este sistema de cifrado para este caso, ya que guardaríamos las claves en la base de datos de forma cifrada, y cuando tengamos que autenticar a un usuario deberíamos cifrar la contraseña enviada por él y comprobar que coincide con la que tenemos en la base de datos.

Tendremos que estar muy atentos siempre a estos temas de seguridad en el módulo de comunicación, debido al tráfico de datos confidenciales que se produce a través de la red y a los posibles ataques que se puedan producir contra el sistema.

7.2. Optimización del rendimiento del proyecto

En este apartado, mostraremos algunas de las posibles optimizaciones que se pueden llevar a cabo sobre el proyecto después de realizar las investigaciones y estudios pertinentes.

Una de las posibles líneas de investigación sería la realización de un análisis de cómo optimizar el tráfico de la red, ya que el sistema global (Sistema de gestión de información turística) da servicio a usuarios de una ciudad o región, con lo que conlleva la existencia de un porcentaje importante de clientes que intentarán beneficiarse de los servicios ofrecidos, provocando un tráfico denso en la red. Una conclusión del análisis podría ser la de sintetizar al máximo el formato de los mensajes, es decir, que no exista información innecesaria dentro de los mensajes para los módulos del sistema que la reciben.

Dentro del proyecto existen otras líneas de trabajo, que podrían estar relacionadas con la optimización del rendimiento de las aplicaciones, como podría ser el caso de optimizar el rendimiento de la librería del módulo de comunicación del dispositivo móvil con la supresión de las operaciones de comprobación de los formatos de los mensajes, realizando esta comprobaciones sólo en el servidor del módulo de comunicación. Esto ayudaría sobre todo a que la aplicación de usuario del dispositivo móvil fuera lo menos pesada posible, favoreciendo así a las ciertas limitaciones de procesamiento que poseen estos dispositivos móviles.

Todas estas líneas de investigación y trabajo relacionadas con la optimización del proyecto irán perdiendo importancia en el momento que los dispositivos y equipos en los que se encuentren desplegados tengan mayores capacidades hardware.

7.3. Gestión de la conexión

La gestión de la conexión en el dispositivo móvil no la realiza el módulo de comunicación desplegado en éste, sino que el usuario del dispositivo tiene que activar la conexión a la red a través de Bluetooh, GPRS… o cualquier otro modo de establecerla. El módulo actualmente sólo comprueba que exista una conexión activa para la ejecución de las peticiones solicitadas. En el apartado 4.3.1.3 se explica cómo se realiza esta comprobación.

En un futuro se podría realizar la gestión de la conexión en el módulo de comunicación para que el usuario del dispositivo no tenga que estar pendiente de que se encuentre conectado.

7.4. Soporte del módulo de comunicación en entorno no Windows

Otra posible línea de trabajo consiste en la adaptación del módulo de comunicación desplegado en el dispositivo móvil para que pueda ejecutarse correctamente en entornos sin el sistema operativo de Microsoft Windows, como puede ser un entorno UNIX.

CAPÍTULO 8. BIBLIOGRAFÍA

8.1. Libros y manuales

Manual de introducción a Microsoft Visual Basic 2005 Express Edition

Jorge Serrano Pérez

Microsoft

2006

¿Cómo escribo mi PFC?

Daniel Borrajo

2003

Programación en Visual Basic .Net

Luis Miguel Blanco

Grupo Eidos

2002

Programación de aplicaciones para Internet con ASP 3

Grupo Eidos

Versión 1.0.0

2000

Professional VB 2005

Bill Evjen, Billy Hollis, Rockford Lhotka, Tim McCarthy, Rama Ramachandran, Kent Sharkey, Bill Sheldon

Wiley Publishing, Inc.

2005

8.2. Sitios Web

http://www.elguruprogramador.com.ar

Portal Web dedicado a la edición de temas relacionados con programación y desarrollo web.

Dentro de este portal se han consultado las siguientes páginas:

http://www.elguruprogramador.com.ar/articulos/aplicaciones-cliente-servidor-en-visual-basic-utilizando-el-control-winsock.htm

En concreto este artículo se expone como debemos crear una aplicación cliente-servidor utilizando como lenguaje de programación Visual Basic.

Fecha de consulta: 20/05/2007

http://www.elguruprogramador.com.ar/articulos/introduccion-al-sql-desde-visual-basic.htm

Este artículo nos presenta como debemos trabajar con SQL utilizando como lenguaje de programación Visual Basic.

Fecha de consulta: 15/07/2007

http://www.opennetcf.com/

El sitio proporciona una introducción a Windows Mobile y al desarrollo de aplicaciones móviles, mediante la propagación de proyecto de código abierto, artículos técnicos y documentos relevantes para el desarrollo de dichas aplicaciones.

Fecha de consulta: 02/06/2007

http://msdn2.microsoft.com/es-es/default.aspx

Esta página pertenece al sitio Web de Microsoft, más concretamente a Microsoft Developer Network (MSDN). Es un recurso muy importante para cualquier desarrollador, arquitecto de software que intente conocer a fondo las tecnologías y productos Microsoft.

Dentro de este portal se han consultado las siguientes páginas:

http://msdn2.microsoft.com/es-es/library/ms172831(VS.80).aspx

Esta página muestra cómo debemos cifrar y descifrar cadenas en Visual Basic.

Fecha de la consulta: 20/06/2007

http://msdn2.microsoft.com/es-es/library/system.security.cryptography.rijndael(VS.80).aspx

Esta página muestra cómo debemos utilizar la clase Rijndael (clase que implementa el algoritmo de cifrado asimétrico Rijndael o AES).

Fecha de la consulta: 21/06/2007

http://msdn2.microsoft.com/es-es/library/3a86s51t(VS.80).aspx

Esta página indica cómo debemos utilizar la instrucción SyncLock para bloquear un bloque de instrucciones.

Fecha de la consulta: 21/08/2007

http://msdn2.microsoft.com/es-es/library/3d6tfc4z(VS.80).aspx

Página que nos enseña como comprobar el estado de la conexión en Visual Basic.

Fecha de la consulta: 16/09/2007

http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/vbref/html/vbsamwindowsformsmultithreadedtciiplistenersample.asp

Página que nos muestra la implementación de una aplicación cliente-servidor utilizando la ejecución Multi-Threaded (en español multi-hilo o multiproceso)

Fecha de la consulta: 25/07/2007

http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/cpref/html/frlrfsystemnetsocketstcpclientclasstopic.asp

Esta página Web nos enseña la utilización de la clase TcpClient mediante el empleo de un ejemplo.

Fecha de la consulta: 25/05/2007

http://msdn2.microsoft.com/es-es/library/system.net.sockets.networkstream.read(VS.80).aspx

Esta página Web nos introduce en la utilización del método Read de la clase NetwoekStream mediante la explicación de un ejemplo.

Fecha de la consulta: 25/05/2007

http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/vsintro7/html/vbtskCreatingInstallerForYourApplication.asp

Este sitio nos enseña como implementar un instalador para una aplicación de Windows.

Fecha de la consulta: 08/09/2007

http://msdn2.microsoft.com/es-es/library/ex526337(VS.80).aspx

La página nos muestra cómo debemos diseñar un sitio Web utilizando ASP .Net.

Fecha de la consulta: 25/07/2007

http://msdn2.microsoft.com/en-us/library/ms834669.aspx

Esta página hace referencia a la API de Bluetooth que posee .Net Framework.

Fecha de la consulta: 20/08/2007

http://msdn.microsoft.com/library/spa/default.asp?url=/library/SPA/cpguide/html/cpconGeneratingKeysForEncryptionDecryption.asp

Esta página nos enseña cómo se realiza un descifrado de datos utilizando tanto el descifrado simétrico como el descifrado asimétrico.

Fecha de la consulta: 21/06/2007

http://msdn2.microsoft.com/es-es/library/xdt4thhy(VS.80).aspx

La página nos detalla como implementar la autenticación de formularios simples en las aplicaciones Web desarrolladas en ASP .Net.

Fecha de la consulta: 05/08/2007

http://support.microsoft.com

Esta página pertenece al sitio Web de Microsoft, más concretamente a ayuda y soporte técnico de Microsoft. Es un recurso interesante, ya que nos ofrece ayuda a la hora de resolver problemas sobre sus tecnologías y productos.

Dentro de este portal se han consultado las siguientes páginas:

http://support.microsoft.com/kb/821770/es

Esta página muestra como determinar el estado de la conexión a Internet utilizando Visual Basic.

Fecha de la consulta: 16/09/2007

http://support.microsoft.com/kb/821768/es

Esta página muestra cómo crear una aplicación cliente-servidor utilizando Visual Basic.

Fecha de la consulta: 20/05/2007

http://support.microsoft.com/kb/821766/es

Esta página nos indica como tenemos que crear un instalador utilizando Visual Studio .Net o Visual Basic 2005.

Fecha de la consulta: 07/09/2007

http://support.microsoft.com/kb/307353/es

Esta página nos muestra como tenemos que crear un paquete de instalación utilizando Visual Studio .Net.

Fecha de la consulta: 07/09/2007

http://support.microsoft.com/kb/308157/es

La página nos detalla como implementar la autenticación basada en formularios de una aplicación ASP.NET mediante Visual Basic .NET.

Fecha de la consulta: 05/08/2007

http://www.fx-soft.com.ar/index.php?option=com_content&task=view&id=15&Itemid=36

Portal Web que nos ofrece una gran cantidad de recursos para programadores, software, tutoriales y consejos para aprender a programar. En concreto este articulo trata de cómo crear una aplicación cliente-servidor en Visual Basic.

Fecha de la consulta: 20/05/2007

http://www.netveloper.com/contenido2.aspx?IDC=166_0

Sitio Web dedicado a tratar temas relacionados con la programación en la plataforma .Net de Microsoft. Este artículo nos muestra los factores que tendríamos que tener en cuenta para empezar en el desarrollo de aplicaciones en dispositivos móviles.

Fecha de la consulta: 18/05/2007

http://forums.microsoft.com/MSDN/ShowPost.aspx?PostID=1851992&SiteID=1

Foros de consulta y discusión acerca de las tecnologías y productos de Microsoft. En el que los usuarios asisten a preguntas y problemas que se producen en el transcurso de la realización aplicación. En concreto este hilo del foro trata sobre la utilización en Visual Basic del algoritmo de cifrado asimétrico Rijndael o AES.

Fecha de la consulta: 21/06/2007

http://robertoyudice.wordpress.com/2007/05/14/como-crear-un-dll-en-visual-studio/

Blog dedicado a la programación en la plataforma .Net de Microsoft. Este artículo nos muestra cómo debemos realizar la creación de una librería DLL.

Fecha de la consulta: 03/09/2007

http://blogs.msdn.com/anthonywong/archive/2006/03/13/550686.aspx

Blog en el que se muestra la creación de una conexión GPRS utilizando la clase TcpClient perteneciente a .Net Framework.

Fecha de la consulta: 17/08/2007

http://www.desarrollomobile.net/default.aspx

Sitio dedicado al desarrollo de aplicaciones y librerías utilizando .Net Compact Framework, cabría destacar el desarrollo de una librería de Bluetooth por parte de este grupo.

Fecha de la consulta: 18/08/2007

http://www.elguille.info/

Portal Web que nos ofrece una gran cantidad de artículos, tutoriales, manuales e información relacionada con el desarrollo en la plataforma .Net.

Dentro de este portal se han consultado las siguientes páginas:

http://www.elguille.info/NET/ASPNET/tutorialLogin/tutorialLogin.htm

Esta página nos muestra un tutorial para la creación de un sitio Web con autenticación mediante formulario.

Fecha de la consulta: 05/08/2007

http://www.elguille.info/colabora/NET2005/gcuadra_VBSQL2005.htm

Esta página nos muestra cómo podemos sacar el mayor beneficio posible a la combinación de Visual Basic 2005 con SQL Server 2005.

Fecha de la consulta: 15/07/2007

http://www.elguille.info/NET/dotnet/comprobar_usuario_usando_base_datos_vb2003.htm

Esta página nos indica cómo podemos comprobar la identificación de un usuario empleando una base de datos.

Fecha de la consulta: 10/07/2007

http://www.elguille.info/colabora/puntoNET/jmbeas_instaladores/jmbeas_InstaladoresNET.htm

Esta página trata sobre la creación de instaladores para aplicaciones de .Net.

Fecha de la consulta: 07/09/2007

http://www.elguille.info/NET/ADONET/Crear_una_base_de_datos_de_SQL_Server_mediante_codigo_de_Visual_Basic.htm

Este sitio Web nos enseña cómo crear una base de datos mediante código de Visual Basic.

Fecha de la consulta: 06/09/2007

http://www.mistrucos.net/truco-asp-net-2-leer-cadena-conexion-del-webconfig-540.htm

Este sitio Web recoge una gran cantidad de recursos informáticos. En concreto esta página nos muestra cómo podemos acceder a los atributos del archivo “web.config” de una aplicación ASP .Net.

Fecha de la consulta: 28/07/2007

http://thinkingindotnet.wordpress.com/2007/06/17/trucos-creacion-de-programas-de-instalacion-para-aspnet-con-vs-2005/

Blog que nos muestra el procedimiento que hay que seguir para crear un programa de instalación para una aplicación ASP .Net con Visual Studio 2005.

Fecha de la consulta: 08/09/2007

http://forums.microsoft.com/MSDN-ES/ShowPost.aspx?PostID=1114064&SiteID=11

Foros de consulta y discusión acerca de las tecnologías y productos de Microsoft. En el que los usuarios asisten a preguntas y problemas que se producen en el transcurso de la realización de la aplicación. En concreto este hilo del foro trata sobre como detectar la existencia de la conexión GPRS.

Fecha de la consulta: 20/08/2007

http://es.gotdotnet.com/quickstart/aspplus/

Portal Web perteneciente a Microsoft, en que se muestra un tutorial bastante completo sobre el desarrollo de aplicaciones Web mediante la utilización de ASP .Net.

Fecha de la consulta: 25/07/2007

http://developersdotnet.com/blogs/marcos/archive/2007/06/05/criptograf-237-a-algoritmos-asim-233-tricos.aspx

Blog que nos muestra un ejemplo en Visual Basic del empleo de algoritmos de cifrados asimétricos.

Fecha de la consulta: 22/07/2007

http://technet.microsoft.com/es-es/library/ms186312.aspx

Sitio Web perteneciente a Microsoft, que nos detalla paso a paso la creación de una base de datos mediante el uso de la herramienta SQL Server Management Studio.

Fecha de la consulta: 15/07/2007

http://foro.todopocketpc.com/showthread.php?t=109420

Foros de consulta y discusión acerca de todo lo relacionado con los dispositivos móviles. En el que los usuarios asisten a preguntas y problemas que se producen en el transcurso de la realización de la aplicación. En concreto este tema del foro trata sobre la conexión GPRS.

Fecha de la consulta: 20/08/2007

ANEXO A.
INSTALACIÓN Y CONFIGURACIÓN DE LAS HERRAMIENTAS DE DESARROLLO.

1. Instalación de Visual Studio 2005

El procedimiento que deberemos llevar a cabo para la instalación de Visual Studio 2005 es el siguiente:

1. Se introduce el CD-Rom de la aplicación en el lector, y espera a que se inicie automáticamente.

2. Una vez que se muestra la pantalla de instalación de Visual Studio 2005 haga clic en *Instalar Visual Studio2005*.

Figura 42: Instalación Visual Studio 2005 – Paso 2.

3. A continuación, cargará los componentes de instalación, y haga clic en *Siguiente*.

4. Acepte los términos del contrato de licencia, e introduzca la clave del producto y haga clic en *Siguiente*.

5. Seleccione el tipo de instalación, en este caso será, *Instalación personalizada*, y a continuación, haga clic en *Siguiente*.

6. En la siguiente pantalla se muestra los componentes que se desean instalar, en este caso, se marcarán las siguientes opciones.

   La herramienta de lenguaje a marcar será Visual Basic, ya que es el lenguaje que se utilizará en la implementación del proyecto. También, deberá seleccionar la aplicación Microsoft SQL Server 2005 Express x86, ya que es necesaria para la creación de la base de datos.

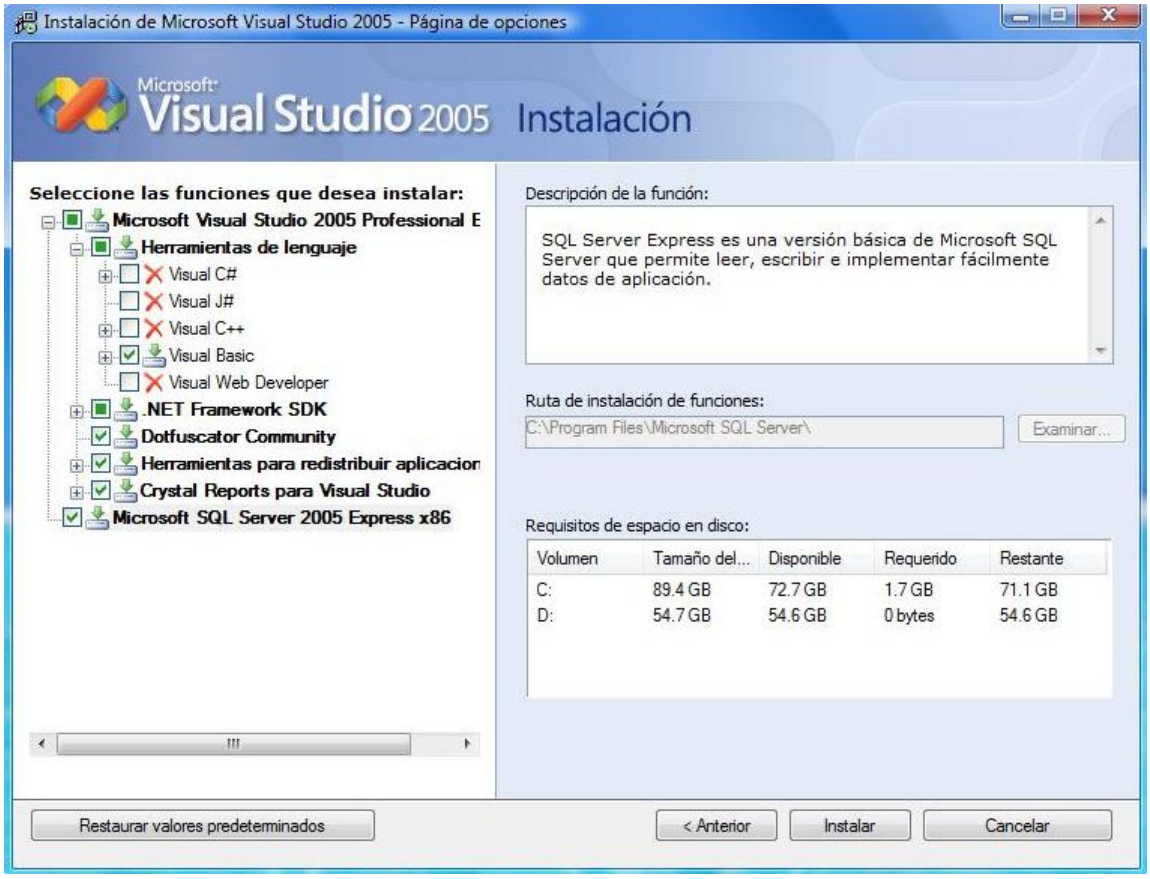


Figura 43: Instalación Visual Studio 2005 – Paso 6.

7. Haga clic en *Instalar*.

8. Si durante la instalación no se produce ningún error, haga clic en *Finalizar*.

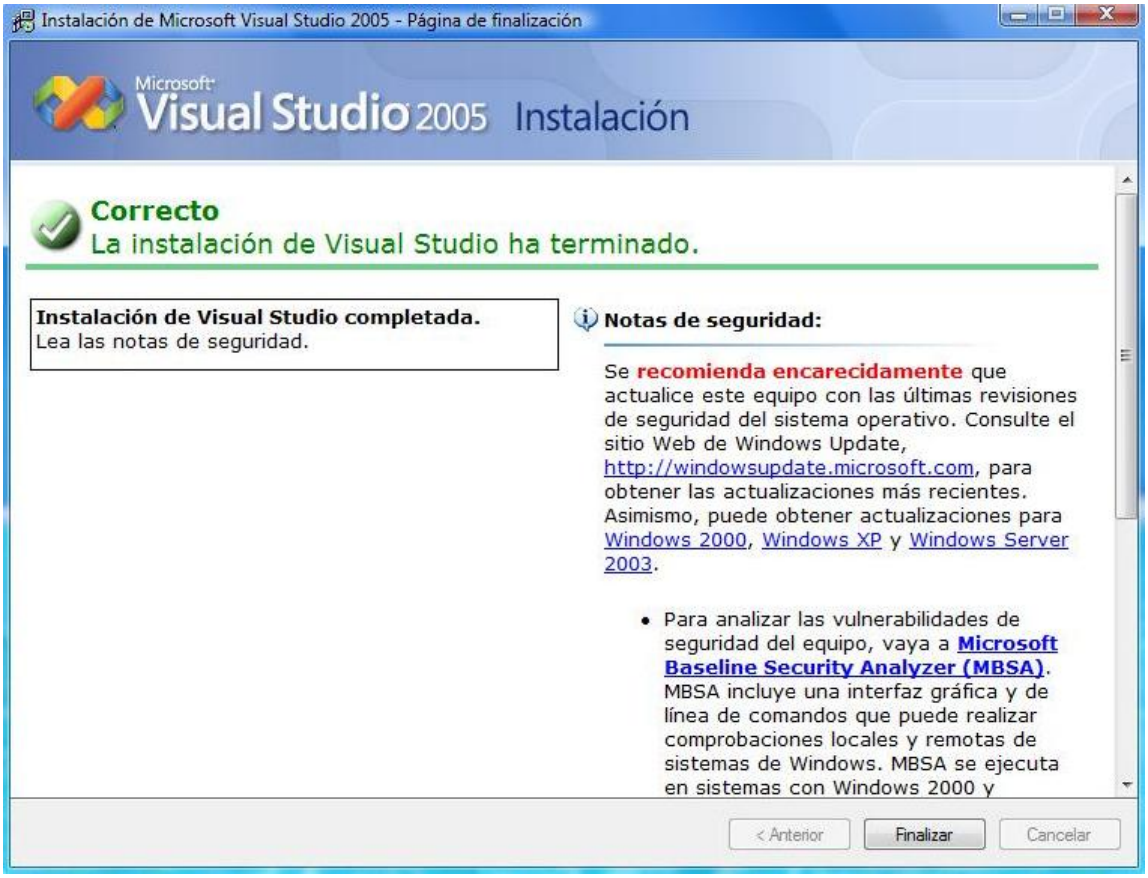


Figura 44: Instalación Visual Studio 2005 – Paso 8.

2. Instalación de Internet Information Services (IIS)

El procedimiento que deberemos llevar a cabo para la instalación de Internet Information Services es el siguiente:

1. Haga clic en *Inicio*, haga clic en *Panel de control*.

2. A continuación, haga clic en Agregar o quitar programas.

3. En Agregar o quitar programas, haga clic en Agregar o quitar componentes de Windows.

4. En el Asistente para componentes de Windows, en la lista Componentes, seleccione Internet Information Services (IIS).

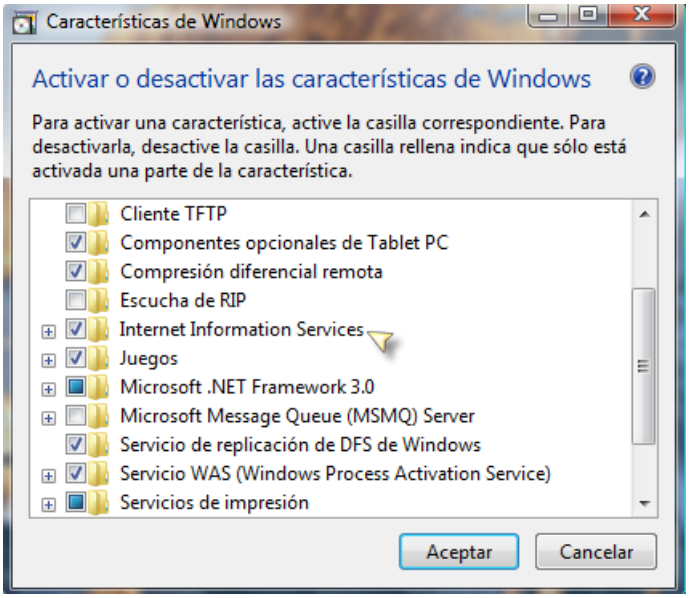


Figura 45: Instalación de Internet Information Services (IIS).

5. Haga clic en *Siguiente*.

6. Cuando el asistente complete la instalación, haga clic en *Finalizar*.

3. Configuración de las herramientas (Visual Studio 2005 y SQL Server 2005)

Lo primero que se debería configurar sería el gestor de la base de datos, el cual debe permitir realizar conexiones de clientes remotos a través de TCP/IP o con canalizaciones de nombre.

Para realizar esta configuración se seguirán los siguientes pasos:

1. Haga clic en *Inicio*, *Archivos de Programas*, *Microsoft SQL Server 2005*, *Herramientas de Configuración*, y haga doble clic en *Configuración de superficie de SQL Server*.

2. Haga clic en *Configuración de superficie para servicios y conexiones*.

Figura 46: Configuración SQL Server – Paso 2.

3. Haga clic en *Conexiones remotas*, que se encuentra en SQLEXPRESS → Database Engine.

4. Seleccione *Conexiones locales y remotas*, y *Usar TCP/IP y canalizaciones con nombre*.

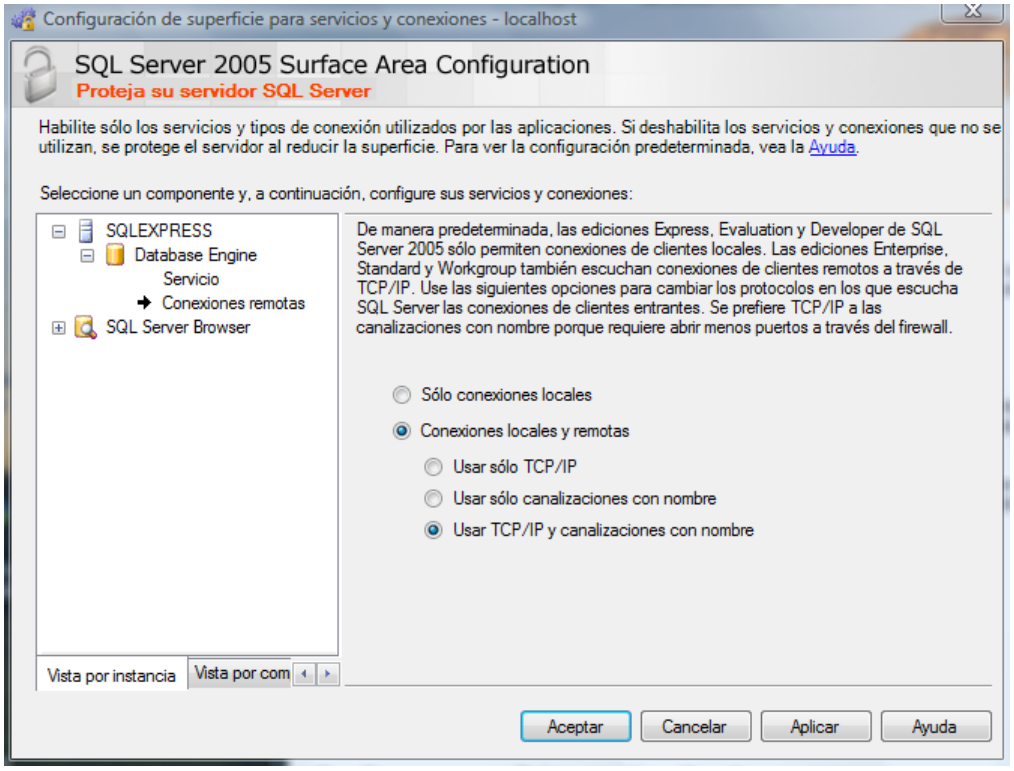


Figura 47: Configuración SQL Server – Paso 4.

5. Haga clic en *Aplicar*, para guardar los cambios realizados, y a continuación a *Aceptar*.

Una vez, terminada la configuración de Microsoft SQL Server 2005, procedemos a la única configuración que tendríamos que realizar en Visual Studio 2005, la que consiste en conectar la herramienta Visual Studio con el gestor de base de datos (SQL Server).

El procedimiento que se tendría que llevar a cabo es el siguiente:

1. Abrir Visual Studio 2005.

2. En la barra de menús, haga clic en *Herramientas* y *Conectar con base de datos…*

3. Se abrirá una nueva ventana para agregar la conexión, en la que hay que seleccionar en primer lugar el Origen de los datos, en este caso, Microsoft SQL Server (SqlClient).

   A continuación, el Nombre del servidor, el cual es, nombreEquipo\SQLEXPRESS.

   Y por último se selecciona el nombre de la base de datos a la que se vaya a conectar, y pulsaremos *Aceptar*, para crear la conexión.

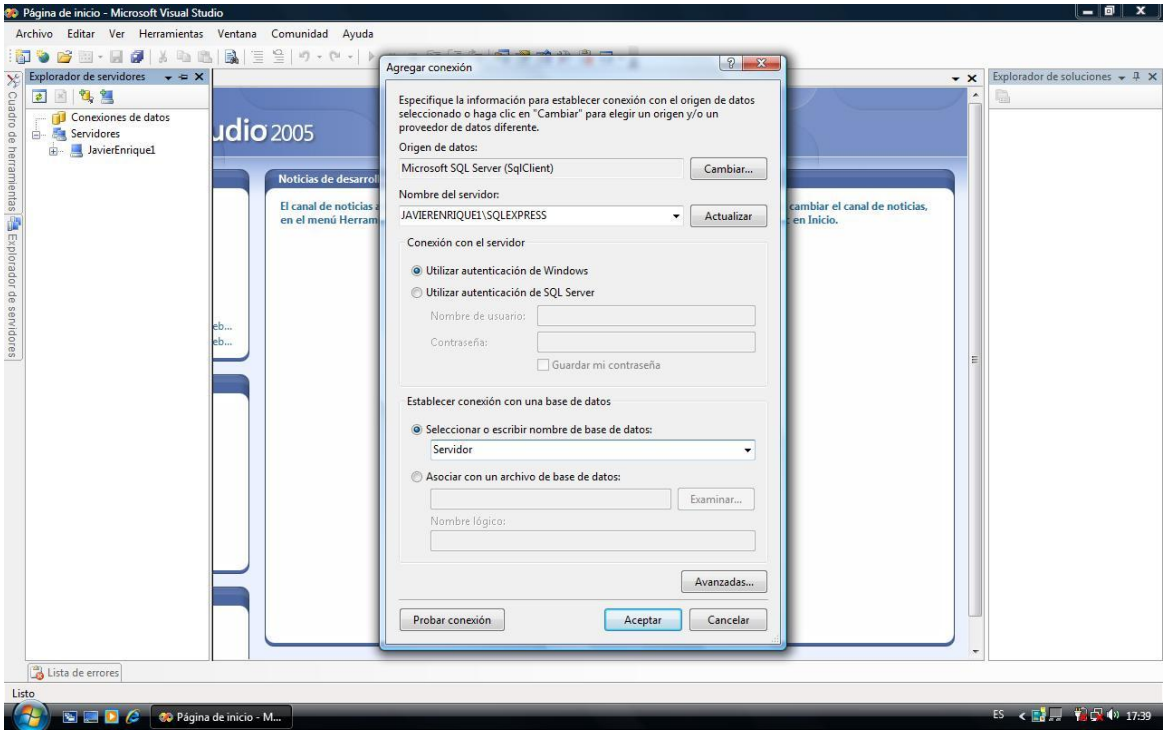


Figura 48: Configuración de Visual Studio 2005.

Para comprobar que la conexión se ha creado correctamente, debe ir a la barra de menús a *Ver* y *Explorador de servidores*. En la ventana que se abra, comprobar que existe la conexión recién creada en *Conexiones de datos*.

Con esto se terminaría la configuración básica de las herramientas de desarrollo.

ANEXO B.
MANUAL DE USUARIO

En este anexo, se presenta el manual de usuario del módulo de comunicación del servidor, en el que explicaremos el proceso de instalación a seguir, la inicialización de las aplicaciones que se realizarán para su correcto funcionamiento y la utilización de las aplicaciones que lo forman (módulo servidor, módulo de administración y módulo de base de datos).

1. Instalación del módulo de comunicación en el servidor

La parte del servidor del módulo de comunicación se distribuye en un archivo autoinstalable. El procedimiento que deberemos llevar a cabo para la instalación del módulo de comunicación del servidor del sistema es el siguiente:

1. Haga doble clic sobre el archivo de "Setup.exe" de la aplicación.
2. A continuación, se abrirá la ventana de instalación, haga clic en *Siguiente*.
3. Seleccione el lugar de la carpeta del disco duro en la que quiere instalar la aplicación, y además, elija si la aplicación es para todos los usuarios que utilicen el equipo o solo para usted. Haga clic en *Siguiente*.

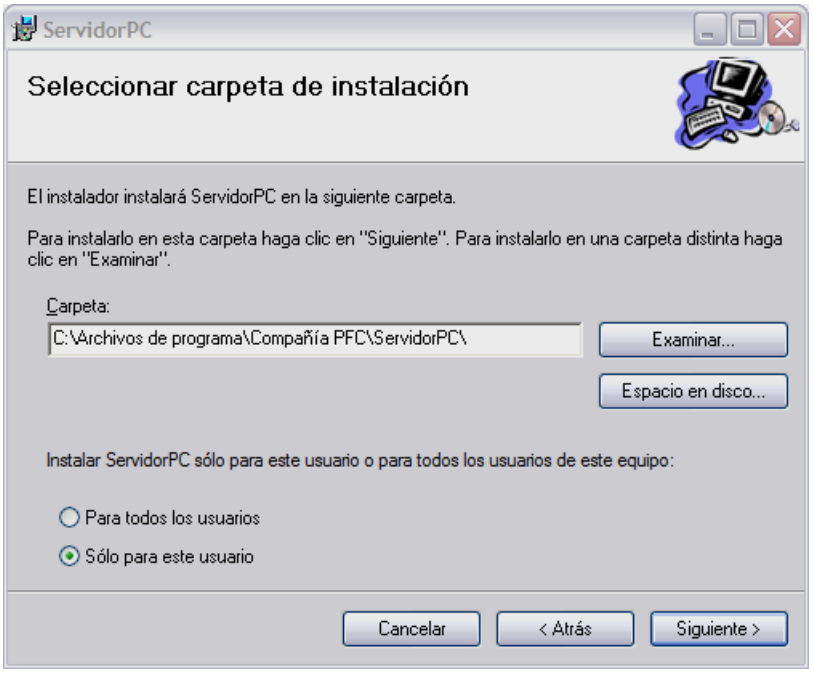


Figura 49: Instalación del proyecto – Paso 3.

4. Vuelva hacer clic en *Siguiente*, para iniciar la instalación.

5. Una vez finalizada la instalación haga clic en *Cerrar*.

Si ha seguido todos estos pasos correctamente, la instalación de la aplicación ha tenido éxito.

2. Desinstalación del módulo de comunicación

Para realizar la desinstalación del módulo de comunicación tendrá que seguir los siguientes pasos:

1. Haga clic en *Inicio*, haga clic en *Panel de control*.

2. A continuación, haga clic en *Agregar o quitar programas*.

3. En *Agregar o quitar programas*, seleccione la aplicación "ServidorPC" y haga clic en *Quitar*.

4. A continuación, se abrirá una ventana en que se le preguntará si desea eliminar la aplicación, en caso de querer eliminarla haga clic en *Sí*, en caso contrario haga clic en *No*.

5. Seguidamente se producirá la desinstalación de la aplicación, que terminará cuando se cierre la ventana.

3. Instalación del módulo de comunicación en la PDA

La parte del módulo de comunicación del dispositivo móvil se distribuye en un archivo DLL, una librería de Windows. El procedimiento que llevaremos a cabo para su utilización en la aplicación de usuario es el siguiente:

1. Importe la librería DLL, "Comunicación", en la aplicación de usuario del dispositivo móvil. Si utiliza Visual Studio 2005 para desarrollar la aplicación usuario los pasos son los siguientes:

    1.1. Abra la solución del proyecto de la aplicación de usuario.

    1.2. Haga clic con el botón derecho del ratón sobre el proyecto en el *Explorador de soluciones* y en el cuadro de diálogo seleccione *Agregar referencia…*

1.3. En la nueva ventana que se muestra entre en la pestaña *Examinar* y seleccione la librería "Comunicaciones", a continuación haga clic en *Aceptar*.

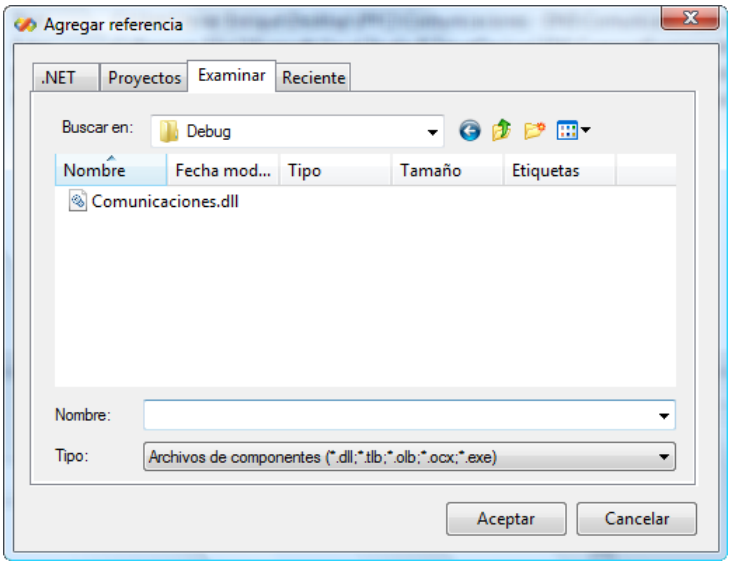


Figura 50: Importar librería Comunicaciones.

2. Cree un objeto de la clase Sesión e invoque el método *Iniciar()*, al que tendrá que pasar como parámetro de entrada el mensaje de la petición. Esta función devolverá el resultado de la petición realizada.

4. Inicialización de las aplicaciones

1. 4.1. Crear la base datos "Servidor"

El procedimiento más sencillo para la creación de la base de datos del módulo de comunicación sería el siguiente:

1. Haga clic en *Inicio*, *Archivos de programas*, *ServidorPC*, y haga clic en *Crear Base de Datos del ServidorPC*.

2. A continuación, se abrirá una ventana para la creación de la base de datos. Si la aplicación detecta que usted no tiene instalado en su equipo Microsoft SQL Server se producirá un error, ya que para poder crear la base de datos debe de tener instalado este gestor.

3. Seleccione la instancia SQL y el nombre de la base de datos a crear. Por defecto, la instancia elegida es "./SQLEXPRESS" y el nombre es "Servidor", utilizadas por el resto de aplicaciones del módulo de comunicación.

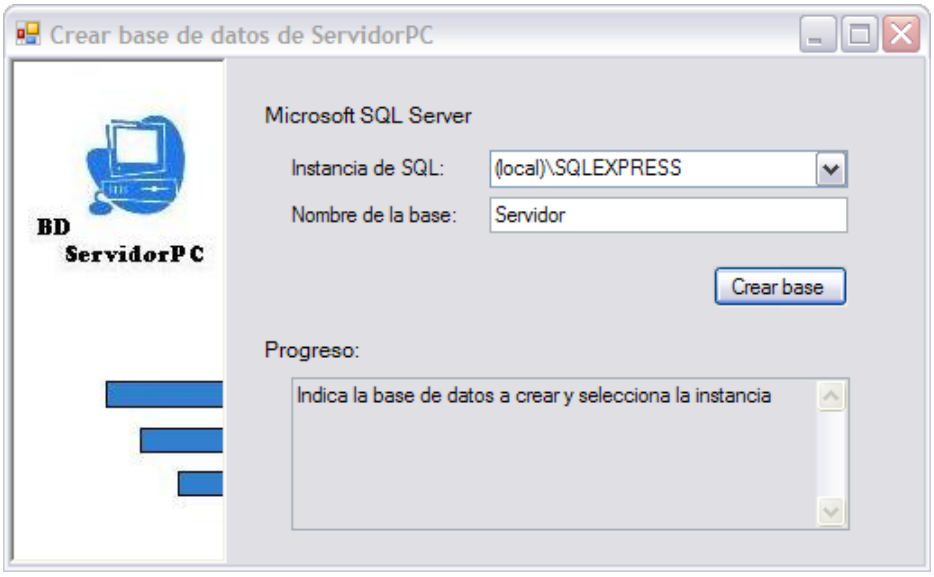


Figura 51: Crear base de datos – Paso 3.

En el caso de que modifique algunos de estos parámetros tendrá que actualizar la cadena de conexión del fichero de configuración del resto de módulos (ServidorPC, Administración). *Ver sección 4.3.2.2.*

4. Haga clic en Crear Base para instanciar la base de datos. Si no se ha producido ningún error durante la operación se mostrará un mensaje de "Base de datos creada correctamente", en caso contrario se indicará cual ha sido el error que se ha producido.

2. 4.2. Publicación del sitio Web de administración

Para la publicación del sitio Web de administración se necesita tener instalado un servidor Web que sea capaz de procesar páginas ASP, ya que el portal Web está formado por páginas implementadas en ASP .Net.

El servidor Web en el que se va a llevar a cabo la publicación es Internet Information Services 7.0 (IIS). Los pasos a seguir son:

1. Haga clic en *Inicio*, haga clic en *Panel de control*.

2. A continuación, haga clic en *Herramientas administrativas*, *Administrador de Internet Information Services (IIS).*

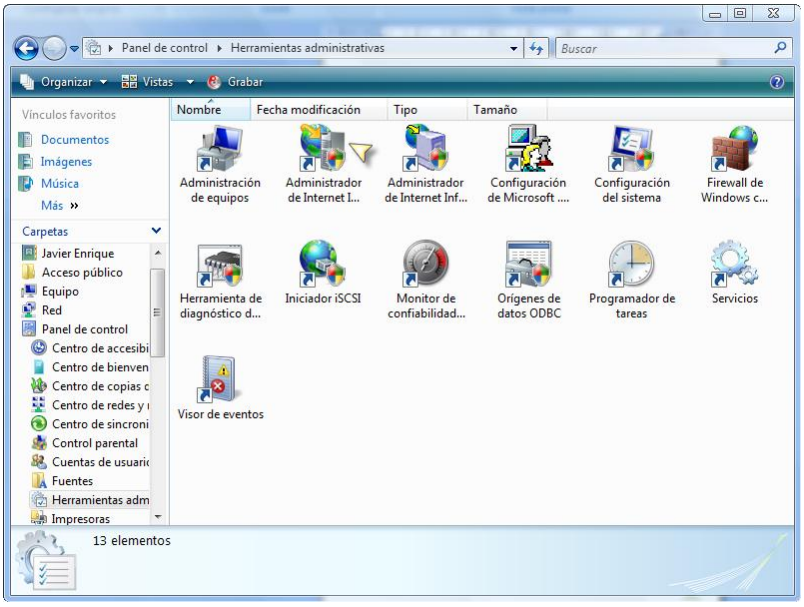


Figura 52: Publicar módulo Administración – Paso 2.

3. Se abrirá una nueva ventana con el Administrador IIS, seleccione la carpeta *Sitios web* del explorador *Conexiones*.

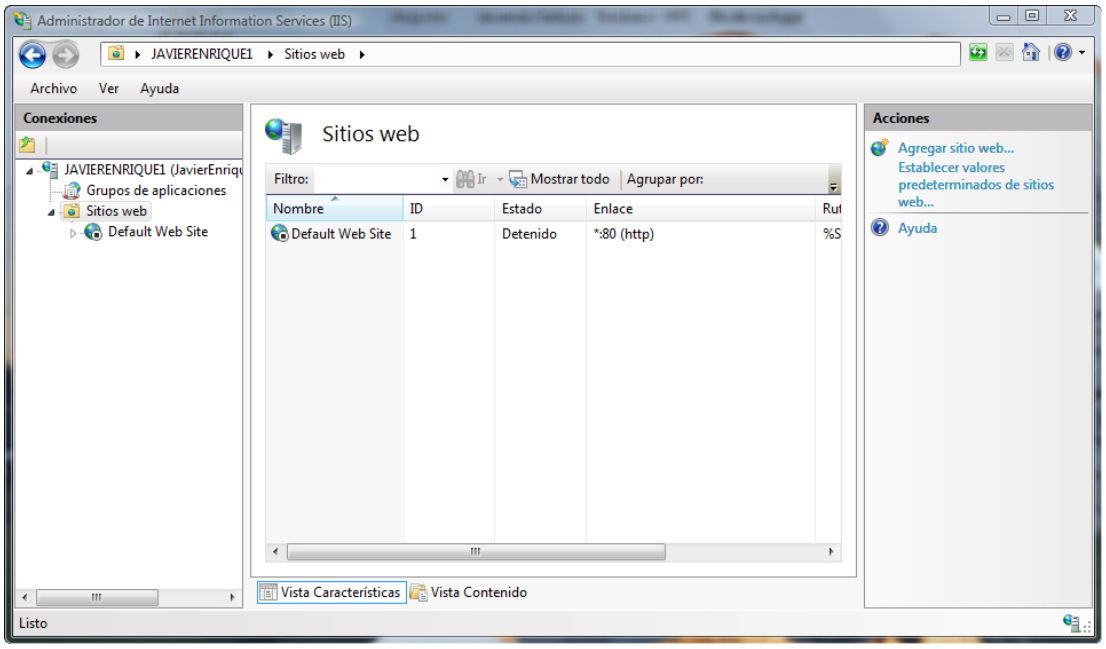


Figura 53: Publicar módulo Administración – Paso 3.

4. Haga clic con el botón derecho del ratón sobre la carpeta *Sitios web*, en el cuadro de dialogo elija *Agregar sitio web…*

5. A continuación, se mostrará una nueva ventana para agregar el sitio web, en la que deberá indicar el nombre y la ruta en la que se encuentra el módulo administración, y asignarle la dirección IP y el puerto. La configuración por defecto sería la siguiente:

   Nombre del sitio web: *Servidor*
   Ruta de acceso física: *C:\Archivos de programa\CompañíaPFC\ServidorPC\Administracion*
   Dirección IP: *Todas las no asignadas*
   Puerto: *80*

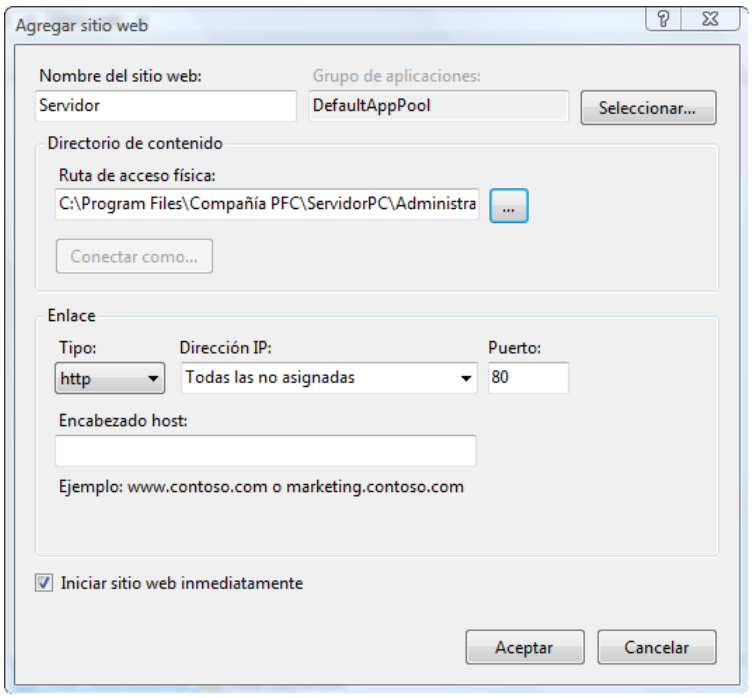


Figura 54: Publicar módulo Administración – Paso 5.

6. Haga clic en *Aceptar* para publicar el sitio web.

Para comprobar que el módulo de Administración funciona correctamente debería abrir el navegador de internet y escribir en la barra de dirección la siguiente URL (Uniform Resource Locator):

http://localhost/login.aspx

En el caso de haber modificado el valor del campo “instancias SQL” o “nombre de la base de datos” o ambos en la creación de la base de datos, tendrá que cambiar la cadena de conexión del módulo administración, lo que se realiza de la siguiente manera:

1. Seleccione el sitio web creado anteriormente, por defecto *Servidor*, y elija la *Cadenas de conexión* de la *Página principal del Servidor*.

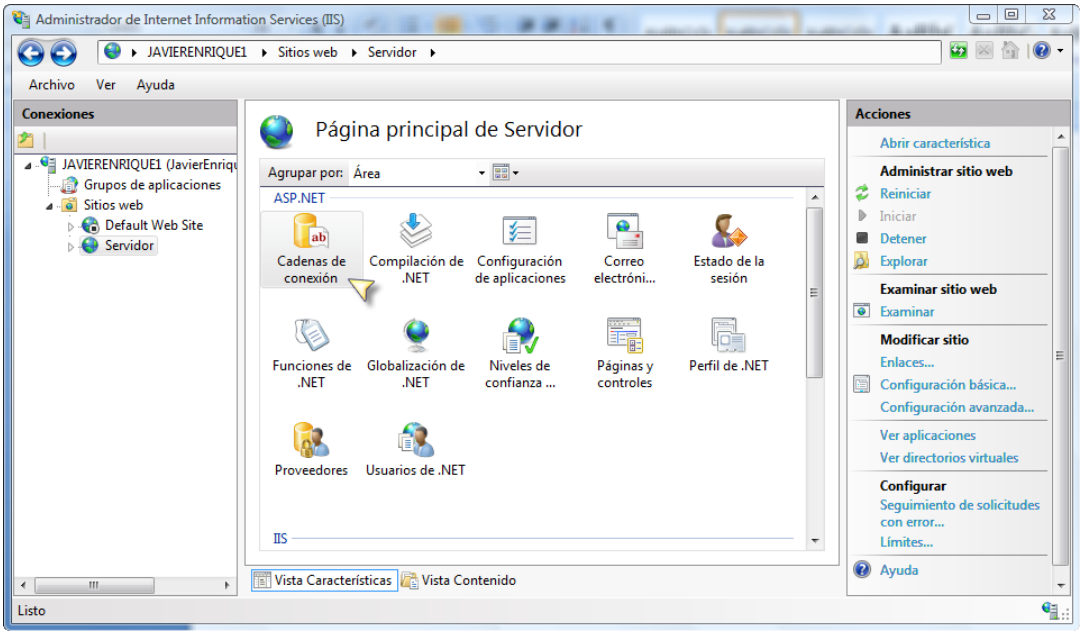


Figura 55: Configuración módulo Administración – Paso 1.

2. A continuación, modifique la cadena de conexión *conexionSqlServer*, en la que tendrá que hacer clic con el botón derecho y seleccionar *Modificar…*

3. Se abrirá una nueva ventana en la que modificará la cadena de conexión.

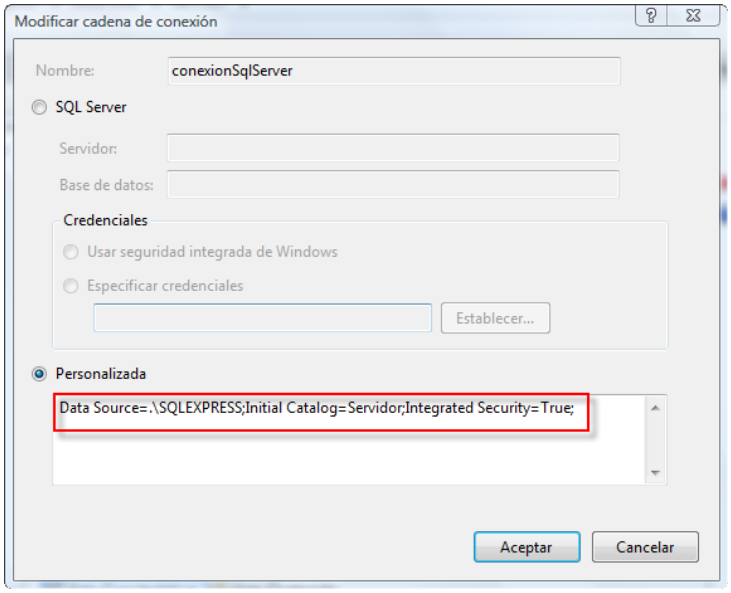


Figura 56: Configuración módulo Administración – Paso 3.

El valor del atributo *Data Source* será modificado por el valor introducido en el campo "Instancias SQL" en la creación de la base de datos. En cambio, sí el campo modificado en la creación de la base de datos ha sido "Nombre de base de datos" el atributo que se debería cambiar es *catalog*. Si los dos campos fueron modificados en la creación de la base de datos ambos deberán ser cambiados.

4. Haga clic en *Aceptar* para guardar la nueva cadena de conexión.

Con esto termina el proceso de publicación del módulo de administración y su configuración, en el caso de que fuera necesaria.

3. 4.3. Configuración del módulo servidor

En este apartado se expondrán los pasos a seguir para la modificación del parámetro de cadena de conexión del archivo de configuración del ServidorPC, en el caso de haber modificado el valor del campo “instancias SQL” o “nombre de la base de datos” o ambos en la creación de la base de datos. Este fichero, “ServidorPC.exe.config”, se encuentra en la carpeta de la aplicación, en el caso de que la ruta de instalación no hubiera sido modificada, el archivo poseería la siguiente ruta:

C:\Archivos de programa\Compañía PFC\ServidorPC\ServidorPC.exe.config

El procedimiento a seguir es:

1. Haga clic con el botón derecho del ratón sobre el fichero de configuración (“ServidorPC.exe.config”), en el cuadro de dialogo seleccione *Abrir con* y a continuación, elija el *Bloc de Notas* como la aplicación que quiere utilizar para abrirlo.

2. Una vez abierto el fichero modificaremos el valor de la etiqueta *connectionStrings* por el introducido en la creación de la base de datos.

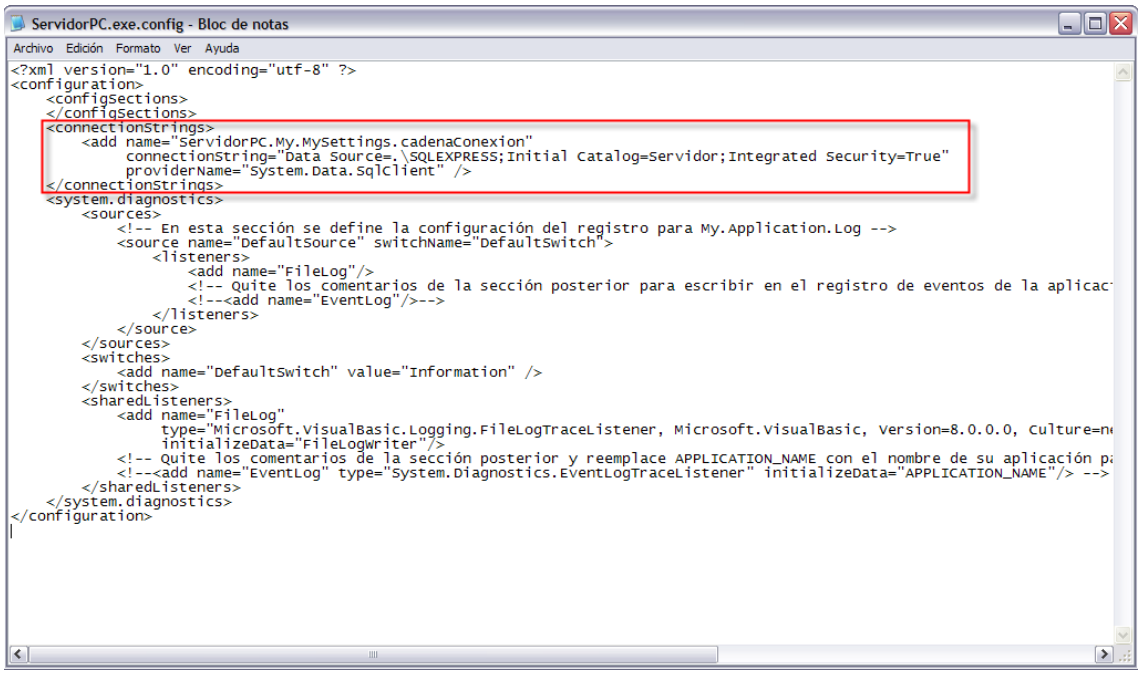


Figura 57: Configurar ServidorPC.

El valor del atributo *connectionString* de la etiqueta *add* será modificado por el valor introducido en el campo "Instancias SQL" en la creación de la base de datos. En cambio, sí el campo modificado en la creación de la base de datos ha sido "Nombre de base de datos" el atributo que se debería cambiar es *catalog*. Si los dos campos fueron modificados en la creación de la base de datos ambos deberán ser cambiados.

3. A continuación, guarde el fichero con las modificaciones realizadas y reinicie el ServidorPC para que el cambio de configuración tenga efecto.

5. Utilización de las aplicaciones

5.1. Sitio Web de administración

El módulo de administración es en el que se llevará a cabo la realización de la gestión de todos los recursos del módulo de comunicación.

Para poder acceder al sitio web de administración necesitará tener permisos de administrador en el módulo de comunicación. Existe un administrador definido en el sistema de forma genérica para que se pueda realizar por primera vez el alta de los nuevos administradores. Los datos de acceso de este administrador genérico son:

Usuario: *adminServidor*
Contraseña: *admin*

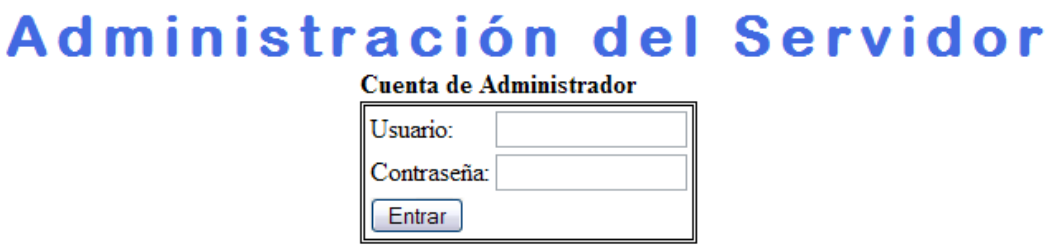


Figura 58: Módulo de Administración – Iniciar sesión.

Después de entrar en el sitio web por primera vez daremos de alta a los primeros administradores del módulo, además de los usuarios y módulos del sistema. Si se requiere se realizaría la actualización de la configuración del ServidorPC.

Figura 59: Módulo de Administración – Pagina principal.

En el caso de que modifique la configuración del ServidorPC o agregue módulos al sistema, el servidor deberá ser reiniciado para que cargue la nueva configuración.

Cuando termine de realizar las gestiones necesarias, cierre la sesión mediante el enlace de Cerrar sesión que se encuentra en la parte superior derecha de la ventana.

5.2. Módulo principal: ServidorPC

La utilización del módulo ServidorPC es muy sencilla, a continuación se expondrá el procedimiento a seguir para iniciar el servidor y saber qué archivo contendrá los errores que se produzcan en el servidor durante su ejecución, en el caso de que existan.

5.2.1. *Iniciar ServidorPC*

Para iniciar el servidor haga clic en *Inicio*, *Archivos de programas*, *ServidorPC*, y haga clic en *Iniciar ServidorPC*. A continuación, se abrirá una ventana en la que se mostrará el puerto y dirección IP donde se está ejecutando el servidor si se ha iniciado correctamente, en caso contrario nos indicará cual ha sido el error que se ha producido.

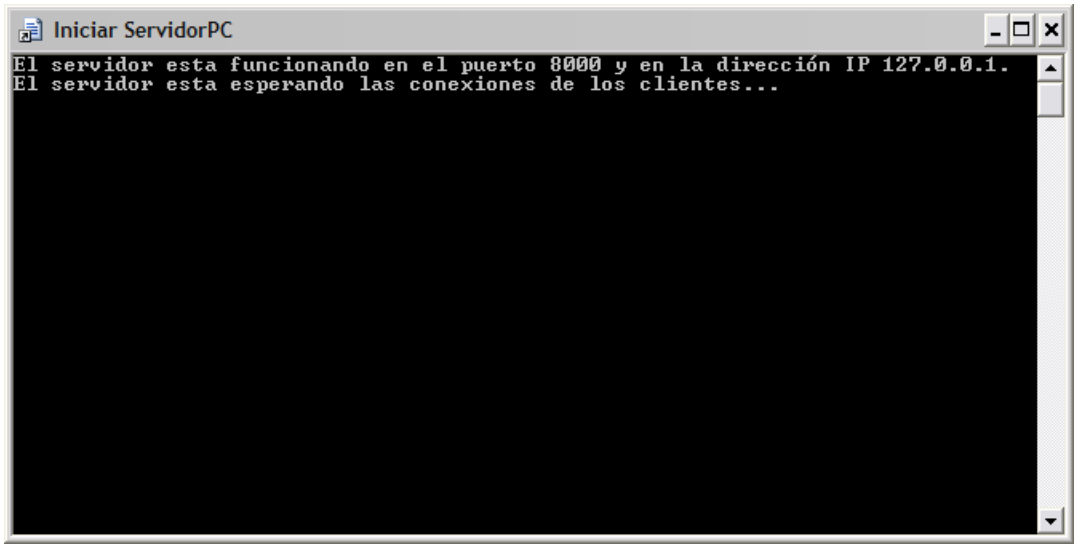


Figura 60: Iniciar ServidorPC.

5.2.2. *Ver archivo de registro de errores "servidor.log"*

El fichero de registro de errores producidos en el ServidorPC se encuentra en la carpeta de la aplicación. Por defecto la ruta del archivo es:

C:\Archivos de programa\Compañía PFC\ServidorPC\servidor.log